UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

Aplicação de Computação Natural ao Problema de Estimação de Direção de Chegada

Autor

Levy Boccato

Orientador

Prof. Dr. Romis Ribeiro de Faissol Attux

Co-Orientador

Prof. Dr. Amauri Lopes

Banca Examinadora:

- Prof. Dr. Romis Ribeiro de Faissol Attux (FEEC/UNICAMP)
- Prof. Dr. Fernando José Von Zuben (FEEC/UNICAMP)
- Prof. Dr. Murilo Bellezoni Loiola (CECS/UFABC)

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica.

Campinas, 12 de Julho de 2010.

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE - UNICAMP

Boccato, Levy
B63a Aplicação de computação natural ao problema de estimação de direção de chegada/ Levy Boccato. – Campinas, SP: [s.n.], 2010.

Orientadores: Romis Ribeiro de Faissol Attux, Amauri Lopes.
Dissertação de Mestrado - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Processamento de sinais. 2. Estimador de máxima verossimilhança. 3. Otimização. 4. Algoritmos evolutivos. 5. Métodos bio-inspirados. I. Attux, Romis Ribeiro de Faissol. II. Lopes, Amauri. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Título em Inglês:	Application of natural computing to the problem of estimating the direction of arrival
Palavras-chave em Inglês:	Signal processing, Maximum likelihood estimator, Optimization, Evolutionary algorithms, Bio-inspired methods
Área de concentração:	Engenharia de Computação
Titulação:	Mestre em Engenharia Elétrica
Banca Examinadora:	Murilo Bellezoni Loiola, Fernando José Von Zuben
Data da defesa:	12/07/2010
Programa de Pós-Graduação:	Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE MESTRADO

Candidato: Levy Boccato

Data da Defesa: 12 de julho de 2010

Título da Tese: "Aplicação de Computação Natural ao Problema de Estimação de Direção de Chegada"

Prof. Dr. Romis Ribeiro de Faissol Attux (Presidente): _____

Prof. Dr. Murilo Bellezoni Loiola: _____

Prof. Dr. Fernando José Von Zuben: _____

Resumo

O problema de estimação de direção de chegada (DOA, em inglês *direction of arrival*) de ondas planas que incidem sobre um arranjo linear uniforme de sensores, através do critério da máxima verossimilhança (ML, em inglês *maximum likelihood*), requer a minimização de uma função custo não-linear, não-quadrática, multimodal e variante com a relação sinal-ruído (SNR, em inglês *signal-to-noise ratio*). Esta dissertação trata da aplicação de algoritmos de computação natural como alternativa ao uso de métodos clássicos, como o MODE e o MODEX, os quais não são capazes de alcançar o desempenho do estimador ML em uma ampla faixa de valores de SNR. As simulações realizadas em diferentes cenários indicam que alguns dos algoritmos analisados conseguem estimar os ângulos de chegada adequadamente. Por fim, inspirados em uma proposta de filtragem de ruído dos dados recebidos, elaboramos uma maneira de realizar a amostragem no espaço de soluções candidatas: a resposta em freqüência do filtro que produz a maior atenuação de ruído é empregada como função densidade de probabilidade no processo de amostragem. Os resultados obtidos atestam que este procedimento tende a aumentar a eficiência dos algoritmos estudados na estimação DOA.

Abstract

The problem of estimating the direction of arrival (DOA) of plane waves impinging on a uniform linear array of sensors, through the maximum likelihood (ML) criterion, requires the minimization of a cost function that is non-linear, non-quadratic, multimodal and variant with the signal-to-noise ratio (SNR). This work deals with the application of natural computing algorithms as an alternative to the use of classical methods, such as MODE and MODEX, which are not capable of achieving the performance of the ML estimator in a wide range of SNR values. The simulations performed in different scenarios indicate that some of the studied algorithms can adequately estimate the angles of arrival. Finally, inspired by a proposal of noise filtering of the received data, we designed a procedure of sampling the search space: the frequency response of the filter which produces the maximal noise reduction is employed as the probability density function during the sampling process. The obtained results attest that this procedure tends to increase the efficiency of the considered algorithms in DOA estimation.

Agradecimentos

Sem dúvida, esta dissertação não é fruto apenas de esforço pessoal. Na verdade, muitas pessoas fizeram parte de todo este árduo processo e, com certeza, merecem toda a minha gratidão por me acompanharem nesta caminhada.

Em primeiro lugar, agradeço a Deus por ter tornado possível a realização deste sonho. Certamente, antes mesmo que eu imaginasse, este momento já estava em Seus soberanos propósitos para minha vida.

Agradeço também aos meus pais, Fernando e Hélia, por todo o carinho, apoio e compreensão ao longo desta jornada. Não há palavras capazes de expressar o quanto sou grato a Deus por tê-los em minha vida. Quando olho para trás, tenho muito orgulho por saber que, somente por meio do esforço deles, eu hoje posso dar este importante passo em minha carreira.

Ao meu irmão, Esdras, por ser meu maior amigo, e por sempre ter me encorajado a avançar nesta direção. À Rebeca, por todo amor, paciência e incentivo demonstrados em todos os momentos.

Aos meus tios, avós e aos demais familiares, por sempre terem estado ao meu lado e também pela ajuda oferecida em várias circunstâncias ao longo deste período.

Aos meus orientadores Romis Ribeiro de Faissol Attux e Amauri Lopes, pela amizade, pelo tempo dedicado e por todo o empenho que demonstraram durante o trabalho. Em cada conversa, ainda que informal, sempre pude aprender com eles lições valiosas para minha carreira e também para minha vida. Ambos mostraram não apenas grande competência, mas também uma incrível compreensão do que é a tarefa de um orientador, e certamente souberam transmitir isto para mim desde os primeiros contatos, durante a iniciação científica.

A todos os amigos do Laboratório de Processamento de Sinais para Comunicações (DSPCom), pela companhia e pelos agradáveis momentos que pudemos desfrutar juntos.

À Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), pela oportunidade concedida e por todo o apoio financeiro.

Sumário

Resumo/Abstract **v**

Agradecimentos **vii**

Abreviaturas **xiii**

Notação Matemática **xv**

Notação de Símbolos **xvii**

Letras Gregas **xxi**

1 Introdução **1**

1.1 Organização da dissertação 3

1.2 Publicações 4

2 Fundamentos de Estimação de Direção de Chegada **7**

2.1 Modelo de Sinal 7

2.2 Estimador de Máxima Verossimilhança 13

2.3 Características da Função ML 17

2.4 Métodos baseados no critério ML 19

2.4.1 O Algoritmo MODE 22

2.4.2 O Algoritmo MODEX 26

2.4.3 Discussão 28
2.5 Conclusão 29

3 Computação Natural **31**
3.1 Introdução 31
3.2 Motivação 33
3.3 Prólogo 35
3.4 Computação Evolutiva 36
3.4.1 Algoritmo Genético 37
3.4.2 Fitness Sharing 43
3.4.3 Fitness Scaling 48
3.4.4 Clearing 49
3.4.5 Estratégias Evolutivas 50
3.4.6 Evolução Diferencial 55
3.5 Sistemas Imunológicos Artificiais 59
3.5.1 CLONALG 61
3.5.2 Opt-aiNet 63
3.6 Inteligência de Enxame 66
3.6.1 Particle Swarm 66
3.6.2 Particle Swarm com fator de inércia 69
3.6.3 Particle Swarm com termo de constrição 70
3.7 Epílogo 71

4 Análise: Estudo de Caso **73**
4.1 Introdução 73
4.2 Testes de Sensibilidade 74
4.2.1 Metodologia 74
4.2.2 Algoritmo Genético 76

4.2.3 Fitness Sharing 80
4.2.4 Fitness Scaling 84
4.2.5 Clearing 87
4.2.6 Estratégias Evolutivas 89
4.2.7 Evolução Diferencial 93
4.2.8 CLONALG 95
4.2.9 Opt-aiNet 98
4.2.10 Particle Swarm 101
4.2.11 Particle Swarm com fator de inércia 103
4.2.12 Particle Swarm com termo de constrição 106
4.2.13 Discussão 108
4.3 Análise de Desempenho 108
4.4 Conclusão 114

5 Análise: Outros Cenários **117**
5.1 Introdução 117
5.2 Outros Cenários 118
5.2.1 Ângulos de chegada distintos 118
5.2.2 Correlação entre fontes 126
5.2.3 Número de fontes 129
5.3 Conclusão 134

6 Uso de Filtragem **137**
6.1 Introdução 137
6.2 Processo de Filtragem 138
6.3 Proposta de uso da filtragem 139
6.4 Resultados 143
6.5 Conclusão 147

7 Conclusões e Perspectivas **149**

7.1 Perspectivas Futuras 151

Referências Bibliográficas **153**

Abreviaturas

AIS	*Artificial Immune Systems* – Sistemas Imunológicos Artificiais
CM	*Conditional Model* – Modelo Condicional
CRB	*Cramér-Rao Bound* – Limite de Cramér-Rao
DE	*Differential Evolution* – Evolução Diferencial
DOA	*Direction Of Arrival* – Direção de Chegada
ES	*Evolution Strategies* – Estratégias Evolutivas
FIR	*Finite Impulse Response* – Resposta ao Impulso Finita
GA	*Genetic Algorithm* – Algoritmo Genético
ML	*Maximum Likelihood* – Máxima Verossimilhança
MODE	*Method Of Direction Estimation*
MODEX	*MODE with EXtra roots*
MUSIC	*Multiple Signals Classification*
PS	*Particle Swarm* – Enxame de Partículas
Radar	*Radio Detection And Ranging*
RMSE	*Root Mean Square Error* – Raiz do Erro Quadrático Médio
SF	*Subspace Fitting* – Ajuste de Subespaço
SNR	*Signal-to-Noise Ratio* – Relação Sinal-Ruído
Sonar	*Sound Navigation and Ranging*
SUS	*Stochastic Universal Sampling*
UM	*Unconditional Model* – Modelo Incondicional

Notação Matemática

a ou A	escalar
$\mathbf{a}$	vetor coluna (letra minúscula em negrito)
$\mathbf{A}$	matriz (letra maiúscula em negrito)
$\mathbf{I}$	matriz identidade
$(\cdot)^*$	complexo conjugado de um escalar, vetor ou matriz
$(\cdot)^T$	transposto de um vetor ou matriz
$(\cdot)^H$	complexo conjugado e transposto de um vetor ou matriz
$\|\mathbf{a}\|$, $\|\mathbf{a}\|_2$	norma 2 do vetor $\mathbf{a}$
$\mathrm{Tr}\{\mathbf{A}\}$	traço da matriz $\mathbf{A}$
$\mathrm{E}\{\cdot\}$	operador esperança
$f(\cdot)$, $g(\cdot)$	função
$\arg\min_{\mathbf{a}}\{f(\cdot)\}$	argumento $\mathbf{a}$ que minimiza a função $f(\cdot)$
$\arg\max_{\mathbf{a}}\{f(\cdot)\}$	argumento $\mathbf{a}$ que maximiza a função $f(\cdot)$
$\sum_i$	somatório com índice i
$\prod_i$	produtório com índice i
$\delta_{k,i}$	operador delta de Kronecker
$\backsim \mathcal{N}(\mathbf{a}, \mathbf{A})$	possui distribuição Gaussiana com média $\mathbf{a}$ e matriz de covariância $\mathbf{A}$
$\backsim \mathcal{U}(a, b)$	possui distribuição uniforme no intervalo (a, b)
$p(\cdot)$	função densidade de probabilidade
$p(\cdot\|\cdot)$	função densidade de probabilidade condicional

$\mathbf{A}^{\dagger}$	pseudo-inversa da matriz $\mathbf{A}$
posto($\mathbf{A}$)	dimensão da imagem da matriz $\mathbf{A}$
$\hat{(\cdot)}$	estimativa de um escalar, vetor ou matriz
$\min(\cdot)$	mínimo valor de $(\cdot)$
$\max(\cdot)$	máximo valor de $(\cdot)$
$\in \mathcal{C}^{A \times A}$	pertence ao espaço complexo de dimensão $A \times A$
$\in \mathcal{R}^{A \times A}$	pertence ao espaço real de dimensão $A \times A$
$\lfloor \cdot \rfloor$	operador piso (*floor*)
$\mathbf{a} \otimes \mathbf{b}$	produto elemento a elemento entre os vetores $\mathbf{a}$ e $\mathbf{b}$
$\binom{a}{b}$	coeficiente binomial
$a \bmod b$	calcula o resto da divisão entre a e b

Notação de símbolos

$\mathbf{A}$, $\mathbf{A}(\boldsymbol{\phi})$, $\mathbf{A}(\boldsymbol{\theta})$	matriz de resposta do arranjo
$\mathbf{Ab}$	matriz com a população de indivíduos (anticorpos)
$\mathbf{Ab_i}$	i-ésimo indivíduo (anticorpo)
$\bar{\mathbf{Ab}}_\mathbf{i}$	matriz com o(s) clone(s) mutado(s) referente(s) ao i-ésimo anticorpo $\mathbf{Ab_i}$
$b(z)$	polinômio de ordem M no domínio transformado z
$\mathbf{b}$	vetor dos coeficientes do polinômio $b(z)$
$\bar{\mathbf{b}}$	vetor $\mathbf{b}$ estendido
$\mathbf{b}_{MODE}$	estimativa MODE do vetor $\mathbf{b}$
$\mathbf{B}$	matriz que descreve a reparametrização da função $J_L(\boldsymbol{\theta})$ para $J_L(\mathbf{b})$
$\bar{\mathbf{B}}$	matriz $\mathbf{B}$ estendida
$\mathbf{C}$	matriz de covariância do vetor $\mathbf{x}_k$
$\mathbf{C}^{(\mathbf{i})}_{\mathbf{Ab}}$	matriz com o(s) clone(s) do i-ésimo anticorpo $\mathbf{Ab_i}$
CR	constante de *crossover*
d	separação (distância) entre os sensores
d_{ik}	distância entre os indivíduos i e k
$E_m(t)$	amplitude do m-ésimo sinal
F	constante que define o passo da mutação
$F_{fit}(\cdot)$, F_{fit}	função de avaliação (*fitness*)
F_{dif}	variação mínima do *fitness* médio da população entre iterações
f_i	valor de *fitness* do i-ésimo indivíduo
$\bar{f}_i$	valor do *fitness* compartilhado do i-ésimo indivíduo

fAb	vetor de *fitness* da população de indivíduos
fPop	vetor de *fitness* da população de indivíduos
fTemp	vetor de *fitness* da população intermediária
fC	vetor de *fitness* dos clones
fAb_i^*	*fitness* do i-ésimo anticorpo normalizado no intervalo $(0, 1)$
H	matriz de filtragem
h	vetor dos coeficientes do filtro FIR
$h(n)$	resposta ao impulso do filtro FIR
$H(z)$	função de sistema do filtro FIR
$H_f(\omega)$	resposta em freqüência do filtro FIR
J_{ML}, $J_{ML}(\boldsymbol{\theta})$	função custo associada ao estimador ML
K	número de *snapshots*
L	ordem do filtro
$L(\cdot)$	versão logarítmica da função de verossimilhança
L_1, L_2	constantes de aceleração
M	número de fontes
$\bar{M}$	número de fontes independentes
M_{max}	número máximo de sinais que podem ser estimados
$\max_{it}$	número máximo de iterações (gerações)
N	número de sensores
N_c	número de clones
N_e	número de experimentos
N_f	número de indivíduos selecionados para *crossover*
N_{fit}	número médio de avaliações da função de *fitness*
N_P	número de pontos amostrados do espaço de busca, número de indivíduos na população
N_t	número de indivíduos que irão competir em um torneio
N_σ	número de direções a serem mutadas
nc_i	número de indivíduos que pertencem ao mesmo nicho que o i-ésimo indivíduo

$\mathbf{Off}$	população de descendentes
P	número de raízes extras no polinômio $b(z)$ estendido
$\mathbf{P_A}$	matriz de projeção da matriz $\mathbf{A}$ (subespaço de sinal)
$\mathbf{P_{\tilde{A}}}$	matriz de projeção ortogonal da matriz $\mathbf{A}$ ou matriz de projeção do subespaço de ruído
$\mathbf{P_B}$	matriz de projeção da matriz $\mathbf{B}$
$\mathbf{Pop}$	matriz com a população de soluções candidatas (indivíduos)
$\mathbf{Pr}$	vetor com as probabilidades de seleção de cada indivíduo
Pr_i	probabilidade de o i-ésimo indivíduo ser selecionado
p_{Ab}	percentual dos anticorpos de menor *fitness* a ser substituído
p_c	probabilidade de *crossover*
$\mathbf{p_g}$	melhor posição já localizada por qualquer membro da vizinhança da i-ésima partícula
$\mathbf{p_i}$	melhor posição já visitada pela i-ésima partícula
p_m	probabilidade de mutação
p_x	precisão de uma grade M-dimensional
$\mathbf{R_y}$	matriz de covariância do vetor $\mathbf{y}_k$
$\hat{\mathbf{R}}_\mathbf{y}$	matriz de covariância estimada do vetor de *snapshot* $\mathbf{y}_k$
$\mathbf{R_{ij}}(\vartheta_{ij})$	matriz de rotação com ângulo ϑ_{ij}
$\mathbf{r}_m$	vetor de direção (*steering vector*) do m-ésimo sinal
range	intervalo de excursão de cada componente da matriz $\mathbf{Pop}$
$\mathbf{S}$	matriz com os desvios padrões σ_m
$s_{n,m}(t)$	sinal da m-ésima fonte no n-ésimo sensor
$sh(d_{ik})$	função de compartilhamento (*sharing*)
$\mathbf{T}$	matriz de rotação
T_{Ab}	período de iterações para a introdução de novos indivíduos
$\mathbf{Temp}$	matriz com a população intermediária
$\hat{\mathbf{U}}_n$	matriz dos autovetores do subespaço de ruído da matriz $\hat{\mathbf{R}}_\mathbf{y}$
$\hat{\mathbf{U}}_s$	matriz dos autovetores do subespaço de sinal da matriz $\hat{\mathbf{R}}_\mathbf{y}$
$\mathbf{u}_{i,G+1}$	*trial vector* – i-ésimo vetor (indivíduo) após recombinação

$\mathbf{v}_{yn}$	vetor posição do n-ésimo elemento do arranjo
$\mathbf{v}_{i,G+1}$	i-ésimo vetor (indivíduo) mutado
$\mathbf{W}$	matriz de simetria
$\mathbf{x}(t)$, $\mathbf{x}(k)$, $\mathbf{x}_k$	vetor das formas de onda dos sinais
$\mathbf{x}_i(t)$	posição da i-ésima partícula na iteração t
$\mathbf{X}$	matriz cujas colunas são os vetores $\mathbf{x}_k$
$y_n(t)$	sinal recebido no n-ésimo sensor
$\mathbf{y}(t)$, $\mathbf{y}(k)$, $\mathbf{y}_k$	vetor dos sinais recebidos pelo arranjo (vetor de *snapshot*)
$\mathbf{Y}$	matriz cujas colunas são os vetores $\mathbf{y}_k$
$\mathbf{z}_k$	vetor de *snapshot* filtrado

Letras Gregas

β_s	fator de *scaling*
$\boldsymbol{\beta}$	vetor composto pelos elementos reais e imaginários do vetor $\mathbf{b}$
$\boldsymbol{\beta}_{MODE}$	estimativa MODE do vetor $\boldsymbol{\beta}$
χ	coeficiente ou termo de constrição
Δ_n	defasagem do n-ésimo elemento do arranjo
ϵ	valor do limite de Cramér-Rao na máxima relação sinal-ruído considerada
ϕ_m	valor verdadeiro do ângulo de chegada da m-ésima fonte
$\boldsymbol{\phi}$	vetor direção de chegada contendo os parâmetros ϕ_m
φ_m	fase arbitrária do m-ésimo sinal no instante $t = 0$
$\boldsymbol{\varphi}$	vetor que denota as estimativas dos ângulos de chegada
$\boldsymbol{\varphi}_{i,G}$	i-ésimo indivíduo da população na iteração (geração) G
$\boldsymbol{\eta}(t)$, $\boldsymbol{\eta}(t)$, $\boldsymbol{\eta}_k$	vetor de ruído nos sensores
κ	capacidade de nicho
λ	número de indivíduos na população de descendentes
λ_m, λ_o	comprimento de onda do m-ésimo sinal e comprimento de onda
$\hat{\mathbf{\Lambda}}_n$	matriz dos autovalores do subespaço de ruído da matriz $\hat{\mathbf{R}}_\mathbf{y}$
$\hat{\mathbf{\Lambda}}_s$	matriz dos autovalores do subespaço de sinal da matriz $\hat{\mathbf{R}}_\mathbf{y}$
$\mathbf{\Lambda}_M$	estimativa (corrigida) da matriz $\hat{\mathbf{\Lambda}}_s$
μ	número de indivíduos na população de pais
$\boldsymbol{\nu_i}(t)$	vetor velocidade da i-ésima partícula na iteração t
ν_{min}, ν_{max}	limites mínimo e máximo das componentes do vetor velocidade

$\boldsymbol{\theta}$	vetor das variáveis de decisão dos parâmetros de direção de chegada
θ_{ij}	estimativa do i-ésimo ângulo de chegada no j-ésimo experimento
$\boldsymbol{\vartheta}$	vetor com os ângulos de rotação
$\vartheta_i, \vartheta_{ij}$	ângulos de rotação
ρ	fator de decaimento da mutação
σ_k^2, σ^2	variância do ruído no k-ésimo *snapshot* e variância do ruído
$\boldsymbol{\sigma_m}$	vetor que define o desvio padrão em cada direção
${\sigma_m}^{(i)}$	desvio padrão na i-ésima direção
σ_{mating}	limiar de similaridade para *crossover*
σ_s	limiar de similaridade entre indivíduos
Ω_m, Ω	freqüência da portadora do m-ésimo sinal e freqüência da portadora
$\boldsymbol{\omega}$	vetor das freqüências angulares relativas aos M ângulos de chegada
ω_m	freqüência angular do m-ésimo sinal relativa ao ângulo ϕ_m
$\boldsymbol{\psi_1}$, $\boldsymbol{\psi_2}$	vetores aleatórios positivos com distribuição uniforme $\mathcal{U}(0, L_1)$ e $\mathcal{U}(0, L_2)$
ζ	peso ou fator de inércia

Capítulo 1

Introdução

O processamento de sinais por arranjo de sensores constitui uma área de pesquisa que tem exercido um papel importante no desenvolvimento de muitos ramos da ciência aplicada. Alguns problemas particularmente relevantes emergem em contextos de localização de fontes, como em aplicações ligadas a Radar (*Radio Detection and Ranging*) e Sonar (*Sound Navigation and Ranging*). No entanto, diversas outras áreas também se beneficiaram com o uso de arranjos de sensores, dentre as quais destacamos telecomunicações, explorações geológicas, radioastronomia e tomografia (Krim e Viberg, 1996) (Haykin, 1985).

Um arranjo de sensores é um conjunto de sensores (transdutores) dispostos segundo uma geometria no espaço tridimensional, com um ponto de referência em comum (Manikas, 2004). A função dos elementos do arranjo é coletar amostras da excitação que o ambiente exerce sobre ele para que seja possível extrair informações espaço-temporais de sinais corrompidos por ruído. As ondas que atingem o arranjo podem ser geradas por fontes de irradiação ou resultam de reflexões de ondas transmitidas. As informações de maior interesse são o número de sinais incidentes, parâmetros destes sinais (como amplitude, fase e freqüência), a velocidade de propagação das ondas e a localização da fonte emissora, a qual é determinada pelo ângulo de chegada ou DOA (*Direction Of Arrival*) (Van Trees, 2001b).

Neste trabalho, nosso interesse se concentra no problema de estimação das direções de

chegada de sinais incidentes no arranjo de sensores. Além disto, consideramos a abordagem paramétrica de estimação baseada no critério de máxima verossimilhança (em inglês, *Maximum Likelihood* (ML)). Esta abordagem se mostra adequada e com desempenho igual ou superior ao de outras, como a abordagem baseada no critério de ajuste de subespaços (SF, em inglês *Subespace Fitting*), especialmente em condições críticas de relação sinal-ruído (SNR, em inglês *signal-to-noise ratio*) e número de *snapshots* (Krim e Viberg, 1996).

Segundo o critério ML, a estimativa de um parâmetro é obtida a partir da maximização da função de verossimilhança, a qual, no caso do problema de estimação DOA, possui um caráter não-linear, não-quadrático e multimodal, especialmente em condições críticas de relação sinal-ruído (Van Trees, 2001b) (Krummenauer, 2007). Além disso, a superfície desta função varia consideravelmente à medida que a SNR é alterada, de maneira que a posição de seus pontos modais pode ser bastante modificada. Por conta destes fatos, somente uma abordagem de busca exaustiva é capaz de garantidamente localizar o ótimo global, e, portanto, de implementar o estimador ML com perfeição. Entretanto, por ser demasiadamente custosa, particularmente quando o número de sinais incidentes no arranjo aumenta, sua aplicação prática fica inviabilizada.

Diante destes obstáculos, foram propostos métodos alternativos para realizar a estimação dos ângulos de chegada, dentre os quais, destacamos o MUSIC (Schmidt, 1981) (Schmidt, 1986), o MODE (Stoica e Sharman, 1990) e o MODEX (A. Gershman e Stoica, 1999). Contudo, todas estas abordagens apresentam uma degradação de desempenho em relação ao estimador ML à medida que a SNR é reduzida. Em outras palavras, elas também não são capazes de implementar adequadamente o estimador ML.

Nos últimos anos, uma classe de ferramentas computacionais denominadas meta-heurísticas tem conquistado um significativo interesse na comunidade científica, uma vez que oferece alternativas bastante interessantes para lidar com problemas de otimização com características desafiadoras. Embora não sejam capazes de garantir a obtenção da solução ótima, estas técnicas têm o atrativo de poderem ser utilizadas justamente em situações para as quais não foram

concebidos algoritmos exatos de solução ou quando os métodos convencionais que efetivamente encontram a melhor solução são infactíveis.

Por estes motivos, podemos dizer que as meta-heurísticas se credenciam como boas candidatas para resolver o problema de estimação DOA e sua aplicação deve ser investigada.

Neste trabalho, estudamos a aplicação de um conjunto de meta-heurísticas populacionais, mais especificamente, de algoritmos pertencentes à área de pesquisa denominada Computação Natural, ao problema de estimação DOA. Temos por objetivo analisar o desempenho destes algoritmos quando tentam localizar o ótimo global da função de verossimilhança, a fim de verificar se são capazes de implementar o estimador ML.

Convém mencionar que alguns passos foram dados nesta direção, como em (Sharman, 1988), (Sharman e McGlurkin, 1989), (Stoica e Gershman, 1999) e (A. B. Gershman e Stoica, 2000). Neste sentido, este trabalho dá prosseguimento a esta linha de pesquisa ao ampliar o repertório de algoritmos estudados, além de empreender um estudo detalhado sobre os seus principais parâmetros, e também avaliar o desempenho de cada ferramenta na estimação dos ângulos de chegada considerando diferentes cenários DOA.

Por fim, este trabalho introduz uma proposta inovadora, baseada em um processo de filtragem de ruído, para realizar a amostragem do espaço de soluções candidatas de maneira mais inteligente, partindo da informação presente na resposta em freqüência do filtro otimizado, isto é, daquele que produz a maior atenuação do ruído presente nos dados coletados, referente aos valores verdadeiros dos ângulos de chegada.

1.1 Organização da dissertação

O conteúdo desta dissertação está organizado da seguinte maneira:

- Capítulo 2: são apresentados os aspectos básicos do problema de estimação de direção de chegada usando um arranjo linear de sensores uniformemente espaçados. Inicialmente, descrevemos o modelo de sinal e as principais suposições estatísticas para o modelo. Em

seguida, derivamos a expressão referente ao estimador de Máxima Verossimilhança (ML) e ilustramos as características da função a ser otimizada. Por fim, alguns métodos clássicos de estimação são brevemente apresentados e analisados.

- Capítulo 3: primeiramente, destacamos a motivação para se utilizar algoritmos de computação natural como alternativas aos métodos clássicos de estimação DOA. Em seguida, cada algoritmo empregado neste trabalho é apresentado em detalhes.

- Capítulo 4: aqui, temos as primeiras contribuições deste trabalho: 1) a descrição completa do processo de ajuste dos parâmetros dos algoritmos de computação natural. 2) os resultados obtidos em um cenário DOA consagrado na literatura credenciam alguns algoritmos como alternativas viáveis e de excelente desempenho.

- Capítulo 5: a análise do desempenho dos algoritmos de computação natural na estimação dos ângulos de chegada é estendida para outros cenários DOA.

- Capítulo 6: inicialmente, são apresentados os fundamentos do processo de filtragem do ruído presente nos dados recebidos, bem como as características do filtro que maximiza a relação sinal-ruído média em sua saída. Em seguida, apresentamos em detalhes uma proposta inovadora de uso desta filtragem em conjunto com os algoritmos de computação natural.

- Capítulo 7: as conclusões gerais sobre o trabalho e as perspectivas de trabalhos futuros são apresentadas neste capítulo.

1.2 Publicações

Durante o período em que se desenvolveu o mestrado, foram publicados os seguintes trabalhos:

- L. Boccato, F. O. de França, R. Krummenauer, R. Attux, F. J. Von Zuben e A. Lopes, "Otimização Dinâmica Imuno-Inspirada para o Problema de Estimação de Direção de Chegada", Anais do XXVII Simpósio Brasileiro de Telecomunicações (SBrT 2009), 2009.

- L. Boccato, R. Krummenauer, R. Attux e A. Lopes, "Um Estudo da Aplicação de Algoritmos Bio-inspirados ao Problema de Estimação de Direção de Chegada", Revista Controle & Automação, vol. 20, no. 4, p. 609-626, 2009.

Resumo do trabalho - ***Otimização Dinâmica Imuno-Inspirada para o Problema de Estimação de Direção de Chegada*****:** O problema de estimação dos ângulos de chegada (DOA) de sinais incidindo em um arranjo de sensores, mediante o método de máxima verossimilhança, requer a otimização de uma função custo não-linear, não-quadrática, multimodal e variante com a relação sinal-ruído (SNR). Este trabalho apresenta os resultados obtidos por uma rede imunológica especializada em otimização multimodal e dinâmica, denominada *dopt-aiNet*. Simulações considerando cenários estáticos e dinâmicos foram realizadas. No primeiro caso, os resultados mostram que a *dopt-aiNet* consegue localizar com precisão o ótimo global, superando assim alguns métodos clássicos. No segundo caso, sua capacidade de rastrear o ótimo foi verificada e resultados promissores foram atingidos.

Resumo do trabalho - ***Um Estudo da Aplicação de Algoritmos Bio-inspirados ao Problema de Estimação de Direção de Chegada*****:** A solução clássica para o problema de estimação dos ângulos de chegada (DOA) de sinais incidindo em um arranjo de sensores é a aplicação do método de máxima verossimilhança. Este método leva ao problema de otimização de uma função custo não-linear, não-quadrática, multimodal e variante com a relação sinal-ruído (SNR). Os métodos propostos para tal tarefa, presentes na literatura, falham em uma ampla gama de valores de SNR. Este trabalho apresenta os resultados de um estudo sobre a aplicação de ferramentas pertencentes à computação natural ao problema de estimação DOA. Simulações demonstram que quatro dos algoritmos analisados alcançam o ótimo global para

uma ampla faixa de valores de SNR, com esforços computacionais inferiores àquele exigido por uma busca exaustiva.

Capítulo 2

Fundamentos de Estimação de Direção de Chegada

Este capítulo apresenta os fundamentos do problema de estimação de direção de chegada (DOA), definindo assim o alicerce conceitual que será utilizado no restante da dissertação. Além disso, os desafios inerentes a este problema serão evidenciados, de modo a ressaltar a motivação para o estudo da aplicação de ferramentas de computação natural como alternativa aos métodos existentes para estimação DOA.

2.1 Modelo de Sinal

Considere M ondas planas de banda estreita incidindo em um arranjo linear uniforme[1] de N ($N > M$) sensores. Os sensores estão igualmente espaçados e as ondas incidem com ângulos $\boldsymbol{\phi} = [\phi_1 \dots \phi_M]^T$, medidos em relação à normal ao eixo do arranjo, onde $[\cdot]^T$ denota o transposto de um vetor ou matriz. A Figura 2.1 exibe a geometria do arranjo e, em destaque, a m-ésima onda incidente.

O modelo matemático do m-ésimo sinal captado pelo elemento de referência (primeiro sen-

[1]Os conceitos desenvolvidos neste capítulo podem ser adaptados a outras geometrias. Optamos pelo uso do arranjo linear uniforme por conveniência, em razão da simplicidade matemática do seu modelo.

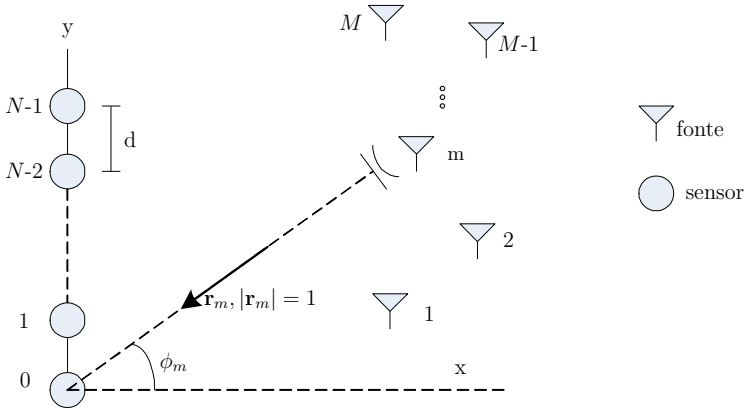


Figura 2.1: Arranjo linear uniforme de N sensores.

sor) do arranjo é dado por (Haykin, 1985) (Krummenauer, 2007)

$$s_{0,m}(t) = E_m(t)e^{j(\Omega_m t+\varphi_m)}, \tag{2.1}$$

onde $E_m(t)$ é a amplitude variante no tempo, Ω_m é a freqüência da portadora, φ_m é uma fase arbitrária no instante $t = 0$ e $m = 1, \ldots, M$.

Considere que a frente de onda associada ao m-ésimo sinal captado esteja se propagando na direção do vetor $\mathbf{r}_m$ com uma inclinação ϕ_m em relação à normal ao eixo do arranjo. Além disso, considere que o vetor posição $\mathbf{v}_{yn}$ do n-ésimo elemento seja tal que $|\mathbf{v}_{yn}| = nd$, onde d corresponde ao espaçamento entre os sensores, e também que λ_m represente o comprimento de onda do m-ésimo sinal captado. Desta forma, podemos expressar a defasagem que cada sinal sofre em relação ao elemento de referência do arranjo através da seguinte expressão (Haykin, 1985):

$$\Delta_n = \frac{2\pi}{\lambda_m}(\mathbf{v}_{yn} \cdot \mathbf{r}_m) = \frac{2\pi}{\lambda_m} nd \sin\phi_m. \tag{2.2}$$

Com o auxílio da Expressão (2.2), podemos escrever a expressão para o m-ésimo sinal

incidente no n-ésimo sensor:

$$s_{n,m}(t) = E_m(t)e^{j(\Omega_m t + \frac{2\pi}{\lambda_m} nd \sin\phi_m + \varphi_m)}. \tag{2.3}$$

Admitimos que todas as ondas planas possuem a mesma freqüência de portadora Ω e que, portanto, o comprimento de onda é o mesmo para todos os M sinais incidentes no arranjo ($\lambda_m = \lambda_o$, $m = 1, \ldots, M$). Além disso, fazemos $d = \frac{\lambda_o}{2}$ para obter máxima resolução (Krummenauer, 2007).

Consideramos também que todas as ondas planas são de faixa estreita, isto é, que $E_m(t)$ tem largura de faixa muito menor que a freqüência de portadora Ω. Admitindo apenas as situações em que o espaçamento entre os sensores é muito menor que a distância entre a m-ésima fonte e o arranjo, podemos supor que $E_m(t)$ é aproximadamente o mesmo em todos os sensores num instante de tempo. Definindo o ângulo de fase elétrica de sensor para sensor como sendo $\omega_m = \frac{2\pi}{\lambda} d \sin\phi_m = \pi \sin\phi_m$, é possível simplificar a Expressão (2.3):

$$s_{n,m}(t) = E_m(t)e^{j(\Omega t + n\omega_m + \varphi_m)}. \tag{2.4}$$

A partir deste ponto, vamos trabalhar com o modelo banda base para o m-ésimo sinal incidente no n-ésimo sensor, desconsiderando assim o termo $e^{j\Omega t}$ da Expressão (2.4), de forma a obter [2]:

$$s_{n,m}(t) = E_m(t)e^{j(n\omega_m + \varphi_m)}. \tag{2.5}$$

Se considerarmos as contribuições de todas as M ondas planas que incidem no n-ésimo sensor e também a presença de ruído aditivo, obtemos o modelo matemático do sinal incidente

[2]Observe que estamos usando a mesma notação para o sinal da Expressão (2.4) e seu equivalente em banda base (Expressão (2.5)).

no n-ésimo sensor de acordo com a Expressão (2.6):

$$y_n(t) = \sum_{m=1}^{M} E_m(t) e^{j(n\omega_m + \varphi_m)} + \eta_n(t), \qquad n = 0, 1, \ldots, N-1. \tag{2.6}$$

Usando notação matricial, reescrevemos a Expressão (2.6) na forma mais compacta

$$\mathbf{y}(t) = \mathbf{A}\mathbf{x}(t) + \boldsymbol{\eta}(t), \tag{2.7}$$

onde $\mathbf{y}(t) \in \mathcal{C}^{N\times 1}$ é o vetor dos sinais recebidos, definido por

$$\mathbf{y}(t) = \begin{bmatrix} y_0(t) \\ y_1(t) \\ \vdots \\ y_{N-1}(t) \end{bmatrix}, \tag{2.8}$$

$\boldsymbol{\eta}(t) \in \mathcal{C}^{N\times 1}$ é o vetor de ruído definido por

$$\boldsymbol{\eta}(t) = \begin{bmatrix} \eta_0(t) \\ \eta_1(t) \\ \vdots \\ \eta_{N-1}(t) \end{bmatrix}, \tag{2.9}$$

$\mathbf{x}(t) \in \mathcal{C}^{M\times 1}$ é o vetor das formas de onda dos sinais definido por

$$\mathbf{x}(t) = \begin{bmatrix} E_1(t)e^{j\varphi_1} \\ E_2(t)e^{j\varphi_2} \\ \vdots \\ E_M(t)e^{j\varphi_M} \end{bmatrix}, \tag{2.10}$$

$\mathbf{A} \in \mathcal{C}^{N\times M}$ é a matriz de resposta do arranjo, definida pela estrutura de Vandermonde

$$\mathbf{A} = \begin{bmatrix} 1 & \dots & 1 \\ e^{j\omega_1} & \dots & e^{j\omega_M} \\ \vdots & & \vdots \\ e^{j(N-1)\omega_1} & \dots & e^{j(N-1)\omega_M} \end{bmatrix}. \tag{2.11}$$

O vetor $\mathbf{y}(t)$ é historicamente denominado *snapshot*, e representa os sinais recebidos pelos sensores em um instante particular t. Em geral, a estimativa DOA é realizada com base em vários *snapshots*, recebidos nos instantes t_k, de modo que a equação do modelo descrita no modo contínuo pode ser expressa na forma discreta:

$$\mathbf{y}_k = \mathbf{A}\mathbf{x}_k + \boldsymbol{\eta}_k, \qquad k = 1, 2, \dots, K \tag{2.12}$$

onde $\mathbf{y}_k = \mathbf{y}(t_k)$ e K é o número total de *snapshots* considerados.

Admitimos que o número M de fontes seja conhecido. Além disso, assumimos que a matriz $\mathbf{A} = \mathbf{A}(\boldsymbol{\phi})$ possui posto cheio em colunas, implicando em $N \geq M$. Outras hipóteses para o modelo da Expressão (2.12) são apresentadas a seguir:

1: O ruído $\boldsymbol{\eta}_k$ é um processo aleatório circular, complexo, Gaussiano e estacionário de média zero, temporal e espacialmente descorrelacionado, e com o momento de segunda ordem

$$E\{\boldsymbol{\eta}_k \boldsymbol{\eta}_i^H\} \;=\; \sigma^2 \mathbf{I} \delta_{k,i}, \tag{2.13}$$

onde $\delta_{k,i}$ é o operador delta de Kronecker, $E\{\cdot\}$ é o operador estatístico esperança, $\{\cdot\}^H$ denota conjugado transposto de uma matriz ou vetor e $\mathbf{I}$ denota uma matriz identidade.

2: O conjunto dos vetores de sinal $\mathbf{x}_k$ (para $k = 1, 2, \dots, K$) é um processo aleatório complexo, Gaussiano e estacionário de média zero, circularmente simétrico,

descorrelacionado temporalmente e com matriz de covariância $\mathbf{C}$. Assim

$$E\{\mathbf{x}_k \mathbf{x}_i^H\} = \mathbf{C}\delta_{k,i}. \tag{2.14}$$

3: Os vetores de sinais $\mathbf{x}_k$ e de ruído $\boldsymbol{\eta}_i$ são descorrelacionados para todo k e i.

É pertinente fazermos uma distinção entre as características estatísticas das fontes de sinal, a partir da qual emergem dois modelos: 1) Modelo Condicional (CM, em inglês *Conditional Model*): a sequência $\{\mathbf{x}_k\}_{k=1}^K$ é sempre a mesma para todos os experimentos dos dados aleatórios $\{\mathbf{y}_k\}_{k=1}^K$. 2) Modelo Incondicional (UM, em inglês *Unconditional Model*): a sequência $\{\mathbf{x}_k\}_{k=1}^K$ varia aleatoriamente de experimento para experimento. Ambos os modelos consideram que $\{\boldsymbol{\eta}_k\}_{k=1}^K$ varia de experimento para experimento.

Se considerarmos o modelo CM, a distribuição dos dados $\{\mathbf{y}_k\}_{k=1}^K$ é dada por $\mathbf{y}_k \backsim \mathcal{N}(\mathbf{A}\mathbf{x}_k, \sigma^2\mathbf{I})$, onde $\backsim \mathcal{N}(\mathbf{m}, \mathbf{C})$ indica que a variável possui distribuição gaussiana de média $\mathbf{m}$ e matriz de covariância $\mathbf{C}$, enquanto que, para o modelo UM, temos $\mathbf{y}_k \backsim \mathcal{N}(\mathbf{0}, \mathbf{R}_\mathbf{y})$, onde

$$\begin{aligned} \mathbf{R}_\mathbf{y} &= E\{\mathbf{y}_k \mathbf{y}_k^H\} \\ &= \mathbf{A}\mathbf{C}\mathbf{A}^H + \sigma^2\mathbf{I}. \end{aligned} \tag{2.15}$$

Para um número finito de amostras (*snapshots*), a matriz de covariância teórica $\mathbf{R}_\mathbf{y}$ pode ser aproximada pela média aritmética

$$\hat{\mathbf{R}}_\mathbf{y} = \frac{1}{K}\sum_{k=1}^{K} \mathbf{y}_k \mathbf{y}_k^H, \tag{2.16}$$

de modo que

$$\lim_{K\to\infty} \hat{\mathbf{R}}_\mathbf{y} = \mathbf{R}_\mathbf{y}. \tag{2.17}$$

2.2 Estimador de Máxima Verossimilhança

Seja $\mathbf{y}_k$ o vetor dos sinais coletados nos N sensores do arranjo para o k-ésimo *snapshot*. A partir destas amostras, desejamos encontrar a estimativa para o vetor direção de chegada ($\boldsymbol{\phi}$) que, com maior probabilidade, tenha gerado os dados amostrados.

O critério da máxima verossimilhança define que o valor ótimo do parâmetro $\boldsymbol{\theta}$ é aquele que maximiza a função de verossimilhança, a qual corresponde à função densidade de probabilidade condicional $p(\mathbf{y}_k|\boldsymbol{\theta})$ [3] (Van Trees, 2001a) (Papoulis e Pillai, 2002).

A função densidade de probabilidade do ruído $p(\boldsymbol{\eta}_k)$, para $\boldsymbol{\eta}_k$ complexo, é dada pela Expressão (2.18).

$$p(\boldsymbol{\eta}_k) = \frac{1}{\pi^N(\sigma^2)^N} \exp\left[-\frac{1}{\sigma^2}\boldsymbol{\eta}_k^H\boldsymbol{\eta}_k\right]. \tag{2.18}$$

Sabemos que transformações lineares e combinações lineares de vetores aleatórios Gaussianos resultam em vetores aleatórios Gaussianos. Portanto, com base na Expressão (2.18), é possível derivar a função densidade de probabilidade de $\mathbf{y}_k$, conforme mostra a Expressão (2.19), para o modelo condicional.

$$p(\mathbf{y}_k|\boldsymbol{\theta}, \mathbf{x}_k, \sigma^2) = \frac{1}{\pi^N(\sigma^2)^N} \exp\left\{-\frac{1}{\sigma^2}[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]^H[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]\right\}. \tag{2.19}$$

Para o caso de múltiplos *snapshots*, é preciso trabalhar com a função densidade de probabilidade conjunta, dada por:

$$p(\mathbf{Y}|\boldsymbol{\theta}, \mathbf{X}, \sigma^2) = \prod_{k=1}^{K} \frac{1}{\pi^N(\sigma^2)^N} \exp\left\{-\frac{1}{\sigma^2}[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]^H[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]\right\}, \tag{2.20}$$

onde $\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \ldots\ \mathbf{x}_K]$ e $\mathbf{Y} = [\mathbf{y}_1\ \mathbf{y}_2\ \ldots\ \mathbf{y}_K]$.

[3] Note que a variável $\boldsymbol{\theta}$ denota para o nosso caso o parâmetro de direção de chegada, enquanto que $\boldsymbol{\phi}$ representa os valores verdadeiros dos ângulos de chegada dos sinais incidentes no arranjo.

A função de verossimilhança em sua versão logarítmica é definida pela Expressão (2.21):

$$L(\boldsymbol{\theta}, \mathbf{X}, \sigma^2|\mathbf{Y}) \propto \ln\{p(\mathbf{Y}|\boldsymbol{\theta}, \mathbf{X}, \sigma^2)\}. \tag{2.21}$$

Substituindo a função densidade de probabilidade conjunta $p(\mathbf{Y}|\boldsymbol{\theta}, \mathbf{X}, \sigma^2)$ na Expressão (2.21), obtemos:

$$L(\boldsymbol{\theta}, \mathbf{X}, \sigma^2|\mathbf{Y}) \propto \ln\left\{\prod_{k=1}^{K} \frac{1}{\pi^N(\sigma^2)^N} \exp\left\{-\frac{1}{\sigma^2}[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]^H[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]\right\}\right\}. \tag{2.22}$$

Conforme destacado anteriormente, o estimador ML é aquele que maximiza a função de verossimilhança. Ou seja, as estimativas ML dos ângulos de chegada estão associadas ao máximo global da função de verossimilhança. Por isso, podemos desprezar os termos constantes presentes na Expressão (2.22), obtendo a expressão matemática final da função de verossimilhança:

$$L(\boldsymbol{\theta}, \mathbf{X}, \sigma^2|\mathbf{Y}) \propto -\left(NK\ln\sigma^2 + \frac{1}{\sigma^2}\sum_{k=1}^{K}[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]^H[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]\right). \tag{2.23}$$

Podemos concentrar[4] a função de verossimilhança para $L(\boldsymbol{\theta}|\mathbf{Y})$ através das estimativas ML de $\mathbf{X} = [\mathbf{x}_1 \ \ldots \ \mathbf{x}_K]$ e σ^2. É possível obter estas estimativas através dos argumentos que satisfazem as equações mostradas a seguir.

$$\frac{\partial L(\boldsymbol{\theta}, \mathbf{X}, \sigma^2|\mathbf{Y})}{\partial \mathbf{x}_k} = \mathbf{0} \quad , \tag{2.24}$$

$$\frac{\partial L(\boldsymbol{\theta}, \mathbf{X}, \sigma^2|\mathbf{Y})}{\partial \sigma^2} = \mathbf{0} \quad . \tag{2.25}$$

[4]O termo concentrar, neste contexto, possui o sentido de eliminar a dependência de um parâmetro desconhecido na função em questão.

Calculando a derivada com relação a $\mathbf{x}_l$ [5], chegamos à seguinte expressão:

$$\frac{\partial L(\boldsymbol{\theta}, \mathbf{X}, \sigma^2|\mathbf{Y})}{\partial \mathbf{x}_l} = -\left(\frac{\partial NK \ln\{\sigma^2\}}{\partial \mathbf{x}_l} + \frac{\partial}{\partial \mathbf{x}_l}\left(\frac{1}{\sigma^2}\sum_{k=1}^{K}[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]^H[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]\right)\right), \quad (2.26)$$

a qual se simplifica para:

$$\sum_{k=1}^{K}\frac{\partial}{\partial \mathbf{x}_l}[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]^H[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k] = \mathbf{0}. \quad (2.27)$$

Sabemos que todas as derivadas em que $l \neq k$ são naturalmente nulas, de modo que basta impor $\frac{\partial}{\partial \mathbf{x}_k}[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]^H[\mathbf{y}_k - \mathbf{A}\mathbf{x}_k] = \mathbf{0}$.

Usando a propriedade de derivação (Brookes, 2005)

$$\frac{\partial(\mathbf{B}\mathbf{x} + \mathbf{c})^H\mathbf{D}(\mathbf{E}\mathbf{x} + \mathbf{f})}{\partial \mathbf{x}} = \mathbf{E}^T\mathbf{D}^T(\mathbf{B}\mathbf{x} + \mathbf{c})^* \quad (2.28)$$

com a devida equivalência entre as matrizes, obtemos a estimativa ML de $\mathbf{x}_k$:

$$\begin{aligned} \hat{\mathbf{x}}_k &= [\mathbf{A}^H\mathbf{A}]^{-1}\mathbf{A}^H\mathbf{y}_k \quad (2.29)\\ &= \mathbf{A}^\dagger\mathbf{y}_k, \end{aligned}$$

onde $\mathbf{A}^\dagger = [\mathbf{A}^H\mathbf{A}]^{-1}\mathbf{A}^H$ é a pseudo-inversa de Moore-Penrose[6] da matriz $\mathbf{A}$. Expressando sob uma forma mais compacta, obtemos

$$\hat{\mathbf{X}} = \mathbf{A}^\dagger\mathbf{Y}. \quad (2.30)$$

[5] A mudança do índice k para l é feita para evitar confusão no cálculo da derivada da Equação (2.24), o qual envolve a derivada de um somatório em k.

[6] Uma boa definição da pseudo-inversa de Moore-Penrose pode ser encontrada em (Van Trees, 2001b).

O próximo passo é calcular a derivada com relação a σ^2:

$$\begin{aligned}
\frac{\partial L(\boldsymbol{\theta}, \mathbf{X}, \sigma^2|\mathbf{Y})}{\partial \sigma^2} &= -\left(\frac{\partial NK \ln\{\sigma^2\}}{\partial \sigma^2} + \sum_{k=1}^{K} \frac{\partial}{\partial \sigma^2} \left(\frac{1}{\sigma^2} [\mathbf{y}_k - \mathbf{A}\mathbf{x}_k]^H [\mathbf{y}_k - \mathbf{A}\mathbf{x}_k] \right) \right) \\
0 &= -\frac{NK}{\sigma^2} + \frac{1}{(\sigma^2)^2} \sum_{k=1}^{K} \|\mathbf{y}_k - \mathbf{A}\mathbf{x}_k\|_2^2 \\
NK &= \frac{1}{\sigma^2} \sum_{k=1}^{K} \|\mathbf{y}_k - \mathbf{A}\mathbf{x}_k\|_2^2. \qquad (2.31)
\end{aligned}$$

Substituindo a estimativa de $\mathbf{x}_k$ na Expressão (2.31), temos:

$$\begin{aligned}
\hat{\sigma^2} &= \frac{1}{NK} \sum_{k=1}^{K} \|\mathbf{y}_k - \mathbf{A}\hat{\mathbf{x}}_k\|_2^2 \\
&= \frac{1}{KN} \sum_{k=1}^{K} [\mathbf{y}_k^H (\mathbf{P}_{\mathbf{A}}^{\perp})^H \mathbf{P}_{\mathbf{A}}^{\perp} \mathbf{y}_k], \qquad (2.32)
\end{aligned}$$

onde $\mathbf{P}_{\mathbf{A}} = \mathbf{A}\mathbf{A}^{\dagger}$ é a matriz de projeção do subespaço de sinal e $\mathbf{P}_{\mathbf{A}}^{\perp} = \mathbf{I} - \mathbf{A}\mathbf{A}^{\dagger}$ é a matriz de projeção do subespaço de ruído. Utilizando as propriedades do operador de projeção[7] na Expressão (2.32), obtemos a seguinte expressão:

$$\hat{\sigma^2} = \frac{1}{NK} \sum_{k=1}^{K} \mathbf{y}_k^H \mathbf{P}_{\mathbf{A}}^{\perp} \mathbf{y}_k. \qquad (2.33)$$

O operador $\mathrm{Tr}\{\cdot\}$ determina o traço de uma matriz e possui a propriedade $\mathrm{Tr}\{a\} = a$, para a escalar. Aplicando esta propriedade à Expressão (2.33), finalmente obtemos a estimativa ML de σ^2:

$$\begin{aligned}
\hat{\sigma^2} &= \frac{1}{NK} \sum_{k=1}^{K} \mathrm{Tr}\{\mathbf{y}_k^H \mathbf{P}_{\mathbf{A}}^{\perp} \mathbf{y}_k\} \\
&= \frac{1}{N} \mathrm{Tr}\{\mathbf{P}_{\mathbf{A}}^{\perp} \hat{\mathbf{R}}_{\mathbf{y}}\}. \qquad (2.34)
\end{aligned}$$

[7] Uma análise mais completa sobre a pseudo-inversa e os operadores de projeção pode ser encontrada em (Van Trees, 2001b).

Substituindo as estimativas de $\mathbf{X}$ e σ^2 na Expressão (2.23), temos a seguinte expressão:

$$L(\boldsymbol{\theta}|\boldsymbol{Y}) \propto -\left(NK \ln\left\{ \frac{1}{N} \mathrm{Tr}\{\mathbf{P}_{\mathbf{A}}^{\perp} \hat{\mathbf{R}}_{\mathbf{y}}\} \right\} + \frac{NK}{\mathrm{Tr}\{\mathbf{P}_{\mathbf{A}}^{\perp} \hat{\mathbf{R}}_{\mathbf{y}}\}} \mathrm{Tr}\{\mathbf{P}_{\mathbf{A}}^{\perp} \hat{\mathbf{R}}_{\mathbf{y}}\} \right). \tag{2.35}$$

Desprezando os termos constantes, obtemos finalmente a expressão da função de verossimilhança para a estimativa de $\boldsymbol{\theta}$ dado $\mathbf{Y}$ dos *snapshots*:

$$L(\boldsymbol{\theta}|\mathbf{Y}) \propto -\ln\left\{ \mathrm{Tr}(\mathbf{P}_{\mathbf{A}}^{\perp} \hat{\mathbf{R}}_{\mathbf{y}}) \right\}. \tag{2.36}$$

Uma vez que a função logarítmica é monotônica, e lembrando que as estimativas ML dos ângulos de chegada são dadas pelo valor de $\boldsymbol{\theta}$ que maximiza a função $L(\boldsymbol{\theta}|\mathbf{Y})$, chegamos enfim a:

$$\hat{\boldsymbol{\theta}}_{ML} = \arg\min_{\boldsymbol{\theta}} J_{ML}(\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \mathrm{Tr}\left\{ \mathbf{P}_{\mathbf{A}}^{\perp} \hat{\mathbf{R}}_{\mathbf{y}} \right\}. \tag{2.37}$$

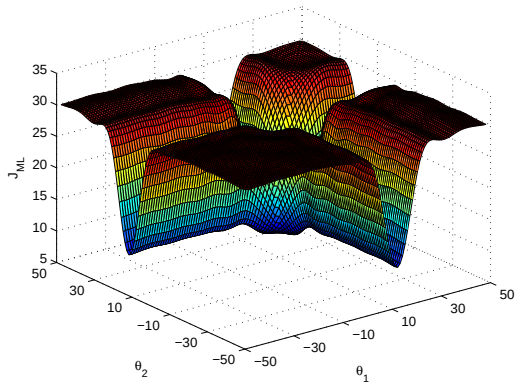
2.3 Características da Função ML

A implementação de algoritmos derivados do critério ML para a estimação dos ângulos de chegada envolve a resolução do problema de minimização da função custo $J_{ML}(\boldsymbol{\theta})$. Esta função é não-linear e não-quadrática em relação ao parâmetro $\boldsymbol{\theta}$, possuindo diversos mínimos locais.

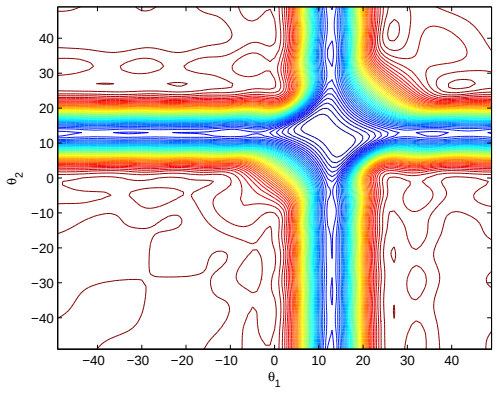
Com o propósito de ilustrarmos as características desafiadoras do problema de minimização da função custo $J_{ML}(\boldsymbol{\theta})$, apresentamos nas Figuras 2.2 e 2.3 as superfícies desta função considerando duas fontes de faixa estreita e de amplitudes unitárias incidindo em um arranjo linear uniforme de 10 sensores com ângulos de chegada $\phi_1 = 10^\circ$ e $\phi_2 = 15^\circ$, para relações sinal-ruído de 0 dB, -5 dB, -10 dB e -15 dB. Foram utilizados $K = 100$ *snapshots*. A matriz de covariância do sinal é dada por $\mathbf{C} = \mathbf{I}$ (sinais descorrelacionados), e a relação sinal-ruído, em dB, é definida como SNR $= 10 \log 1/\sigma^2$, onde σ^2 corresponde à variância do ruído. Os valores dos eixos θ_1 e θ_2 são expressos em graus.

As Figuras 2.2 e 2.3 evidenciam o impacto da relação sinal-ruído na superfície da função

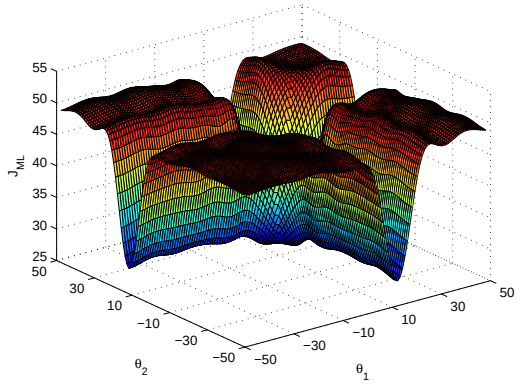
custo $J_{ML}(\boldsymbol{\theta})$. Para uma relação sinal-ruído de 0 dB (Figura 2.2(a)), a superfície tem dois mínimos globais simétricos (localizados, aproximadamente, em $(10°, 15°)$ e $(15°, 10°)$) e um caráter multimodal pouco pronunciado. No entanto, conforme reduzimos a SNR até -15 dB (Figura 2.3(c)), a superfície passa a apresentar diversos mínimos locais. Além disso, o ponto de mínimo global se desloca, de modo que suas coordenadas não mais coincidem com os ângulos de chegada. Estas observações destacam as dificuldades impostas a qualquer técnica baseada no critério ML.



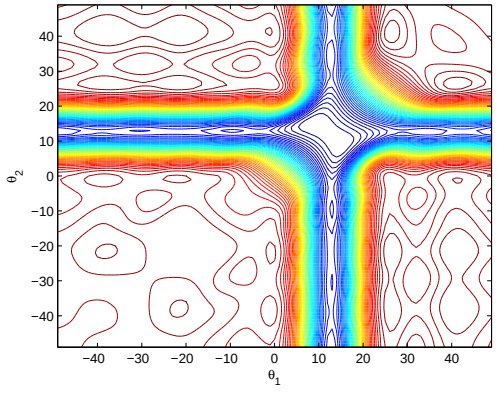
(a) SNR = 0 dB



(b) SNR = 0 dB

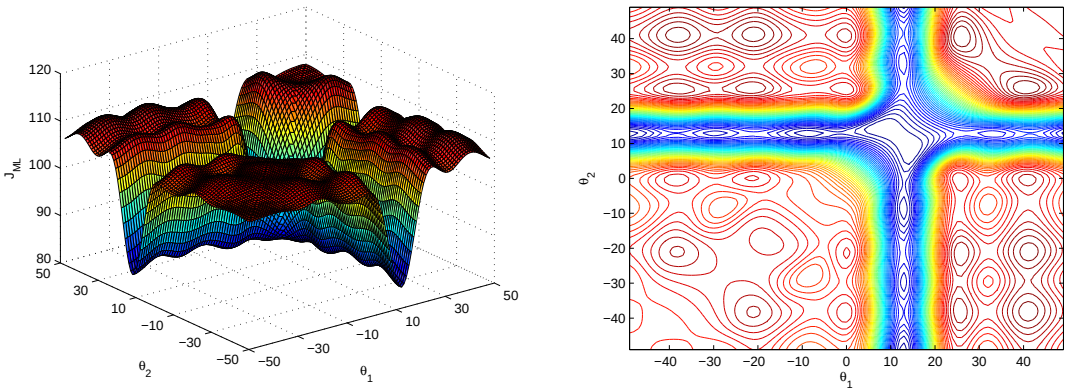


(c) SNR = -5 dB

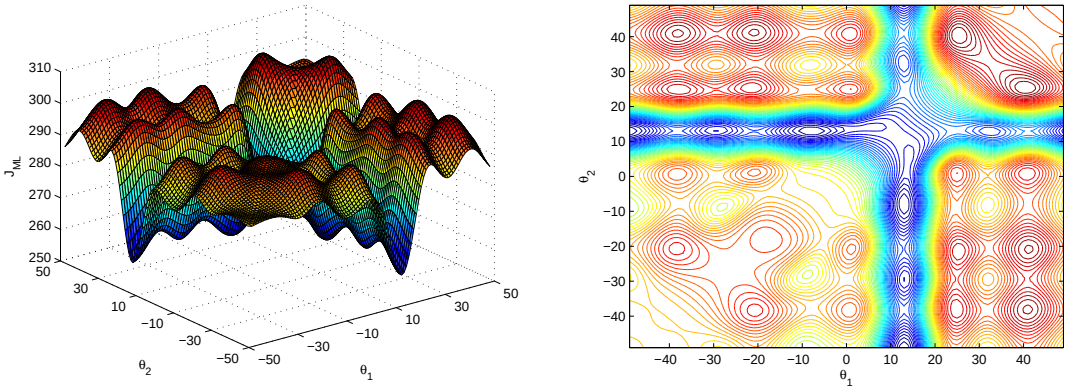


(d) SNR = -5 dB

Figura 2.2: Análise das superfícies e curvas de nível da função ML Condicional para $M = 2$, $N = 10$, 100 *snapshots* e relações sinal-ruído de 0 dB e -5 dB.



(a) SNR = −10 dB (b) SNR = −10 dB



(c) SNR = −15 dB (d) SNR = −15 dB

Figura 2.3: Análise das superfícies e curvas de nível da função ML Condicional para $M = 2$, $N = 10$, 100 *snapshots* e relações sinal-ruído de −10 dB e −15 dB.

2.4 Métodos baseados no critério ML

O critério ML estabelece que as melhores estimativas para os ângulos de chegada estão associadas ao mínimo global da função custo $J_{ML}(\boldsymbol{\theta})$. Na Seção 2.3, explicitamos algumas características desta função que dificultam a identificação do ótimo global. Além disto, observamos o impacto que alterações no valor da relação sinal-ruído podem acarretar à superfície de $J_{ML}(\boldsymbol{\theta})$.

Contudo, há ainda outro fator complicador: uma vez que a relação sinal-ruído não é um

parâmetro conhecido (ou controlado), métodos baseados no critério ML devem ser capazes de lidar com a tarefa de minimizar a função custo $J_{ML}(\boldsymbol{\theta})$ independentemente da SNR. Entretanto, situações como as mostradas nas Figuras 2.2(a) e 2.3(c) configuram problemas de otimização muito distintos. Ou seja, mesmo para um único cenário, como aquele apresentado na Seção 2.3, os métodos ML se deparam com diversos problemas de otimização diferentes, embora todos envolvam a mesma função $J_{ML}(\boldsymbol{\theta})$.

Devido a estas peculiaridades, somente uma abordagem de busca exaustiva no espaço de parâmetros é capaz de garantidamente localizar o ótimo global em todas as circunstâncias, de modo a atingir o desempenho do estimador ML. Contudo, por ser demasiadamente custosa, particularmente quando o número de sinais incidentes no arranjo aumenta, sua aplicação prática fica inviabilizada.

Para fins de análise, podemos fazer uma aproximação da busca exaustiva por meio de uma busca em grade M-dimensional. Supondo uma mesma resolução de grade em cada dimensão, definida pela distância p_x entre dois pontos adjacentes, o número total de pontos existentes na grade, que equivale ao número de vezes que a função custo $J_{ML}(\boldsymbol{\theta})$ será avaliada, é expresso por

$$N_P = \left\lfloor \frac{x_{max} - x_{min}}{p_x} + 1 \right\rfloor^M , \tag{2.38}$$

onde (x_{min}, x_{max}) representa a faixa de excursão em cada dimensão. Note que o número de pontos existentes na grade cresce exponencialmente à medida que o número de fontes M aumenta. Por esta razão, o uso desta abordagem se restringe a cenários com poucas fontes.

Considere o cenário DOA descrito na Seção 2.3. A Figura 2.4 apresenta o desempenho da busca em grade junto com o limite de Cramér-Rao (CRB, em inglês *Cramér-Rao Bound*), o qual estabelece o limite teórico para a variância das estimativas de qualquer estimador não-polarizado (Stoica e Nehorai, 1990).

A métrica utilizada para avaliar a qualidade das estimativas é a Raiz Quadrada do Erro

Quadrático Médio (RMSE), definida como

$$RMSE = \sqrt{\sum_{i=1}^{M}\sum_{j=1}^{N_e}\frac{(\phi_i - \theta_{ij})^2}{MN_e}}, \tag{2.39}$$

onde ϕ_i é o ângulo de chegada do i-ésimo sinal incidente, θ_{ij} é a estimativa do i-ésimo ângulo de chegada no j-ésimo experimento, M é o número de fontes e N_e representa o número de experimentos, isto é, o número de estimativas dos ângulos de chegada que consideramos para uma dada relação sinal-ruído, mantendo o sinal fixo e variando a realização de ruído. Para obtermos valores de RMSE suficientemente confiáveis, utilizamos $N_e = 1000$.

O valor do limite de Cramér-Rao na máxima SNR considerada, o qual denotamos pelo símbolo ϵ, nos dá um indicativo da distância média existente entre o ótimo global da função custo e o ponto associado aos verdadeiros ângulos de chegada. Uma vez que desejamos minimizar os erros introduzidos pela discretização do espaço, feita pela grade bidimensional, devemos utilizar uma grade com resolução $p_x << \epsilon$. Tendo isto em mente, adotamos $p_x = 0,01$. Sendo assim, uma vez que temos $M = 2$, e que $(-90°, 90°)$ indica a faixa de possíveis valores de cada ângulo de chegada, o número de pontos existentes na grade bidimensional é aproximadamente igual a $N_P = 3,2 \cdot 10^8$.

Algumas observações podem ser feitas a partir da Figura 2.4: 1) o desempenho de uma busca em grade, que corresponde a uma aproximação do estimador ML, tende assintoticamente ao limite de Cramér-Rao. 2) à medida que a SNR diminui, tanto o limite de Cramér-Rao quanto o desempenho do método ML pioram; isto porque a ação do ruído faz com que o mínimo global de $J_{ML}(\boldsymbol{\theta})$ se desloque cada vez mais do ponto associado aos valores verdadeiros dos ângulos de chegada, de modo que o erro de estimação cometido é cada vez maior. 3) abaixo de um determinado valor de SNR, denominada SNR de limiar para o ML, a curva RMSE da busca em grade deixa de acompanhar o limite de Cramér-Rao. Este fenômeno é conhecido como efeito de limiar (Rife e Boorstyn, 1974) (Forster, Larzabal, e Boyer, 2004).

A curva RMSE mostrada na 2.4 define a referência de desempenho para os métodos baseados

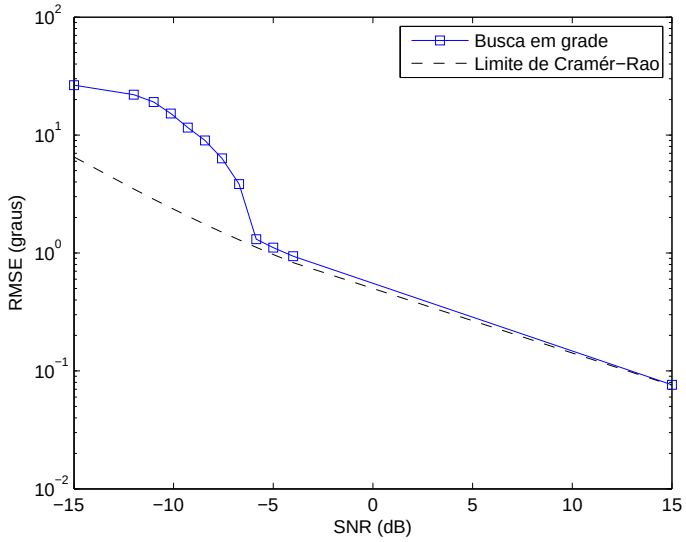


Figura 2.4: Desempenho do método ML em função da SNR.

no critério ML, uma vez que, em todos os experimentos realizados, as estimativas obtidas pela busca em grade correspondem, aproximadamente, ao ótimo global da função custo $J_{ML}(\boldsymbol{\theta})$, e, segundo o critério ML, tal ponto corresponde à melhor estimativa possível para os ângulos de chegada.

O fato de a busca exaustiva não ser factível, como destacado anteriormente, motivou o desenvolvimento de métodos alternativos baseados no critério ML. Estes métodos tentam alcançar o desempenho oferecido pela busca exaustiva, porém com custo computacional inferior. A seguir, apresentaremos os fundamentos de dois destes métodos alternativos, a saber, o MODE e o MODEX.

2.4.1 O Algoritmo MODE

Introduzido por Petre Stoica e K. C. Sharman (Stoica e Sharman, 1990), o algoritmo MODE (*Method of Direction Estimation*) é um estimador baseado no critério ML que emprega uma reparametrização da função custo $J_{ML}(\boldsymbol{\theta})$ e que impõe uma restrição ao subespaço de sinal da matriz de covariância dos *snapshots*.

Reparametrização

Conforme mostrado na Seção 2.2, as estimativas dos ângulos de chegada, segundo o critério ML, estão associadas ao mínimo global de

$$J_{ML}(\boldsymbol{\theta}) = \mathrm{Tr}\{\mathbf{P}_{\mathbf{A}}^{\perp}\hat{\mathbf{R}}_{\mathbf{y}}\}. \tag{2.40}$$

A função $J_{ML}(\boldsymbol{\theta})$ pode ser reescrita em função dos coeficientes b_i do polinômio $b(z)$, definido como

$$b(z) \triangleq b_0 z^M + b_1 z^{M-1} + ... + b_M \triangleq b_0 \prod_{m=1}^{M}(z - e^{j\omega_m}). \tag{2.41}$$

Note que $b(z)$ possui zeros na circunferência de raio unitário dados por $e^{j\omega_1}$, $e^{j\omega_2}$, ..., $e^{j\omega_M}$. Isto significa que

$$\sum_{k=0}^{M} b_{M-k} e^{j(k-q)\omega_m} = 0\ , \qquad m = 1, 2, \ldots, M \tag{2.42}$$

para qualquer inteiro q.

Reescrevendo a Expressão (2.42) em notação matricial, observamos que

$$\mathbf{B}^H\mathbf{A} = \mathbf{0}, \tag{2.43}$$

onde

$$\mathbf{B}^H = \begin{bmatrix} b_M & b_{M-1} & \ldots & b_0 & 0 & 0 & \ldots & 0 \\ 0 & b_M & \ldots & b_1 & b_0 & 0 & \ldots & 0 \\ \vdots & & & & & & & \vdots \\ 0 & 0 & \ldots & b_M & b_{M-1} & & \ldots & b_0 \end{bmatrix}_{(N-M)\times N} . \tag{2.44}$$

Assim sendo, concluímos que as linhas da matriz $\mathbf{B}^H$ (posto $= N - M$) são ortogonais às colunas de $\mathbf{A}$ (posto $= M$), e, portanto, geram o espaço ortogonal da matriz $\mathbf{A}$, o que significa que as matrizes de projeção $\mathbf{P_B}$ e $\mathbf{P}_{\mathbf{A}}^{\perp}$ são equivalentes (Kumaresan, Scharf, e Shaw, 1986)

(Stoica e Sharman, 1990). Ou seja,

$$\mathbf{P}_{\mathbf{A}}^{\perp} = \mathbf{P}_{\mathbf{B}} = \mathbf{B}(\mathbf{B}^H\mathbf{B})^{-1}\mathbf{B}^H. \tag{2.45}$$

Utilizando a identidade entre as matrizes de projeção de (2.45) na expressão da função custo (2.40), obtemos a função custo reparametrizada

$$J_{ML}(\mathbf{b}) = \mathrm{Tr}\{\mathbf{P}_{\mathbf{B}}\hat{\mathbf{R}}_{\mathbf{y}}\} \quad = \mathrm{Tr}\{\mathbf{B}(\mathbf{B}^H\mathbf{B})^{-1}\mathbf{B}^H\hat{\mathbf{R}}_{\mathbf{y}}\}, \tag{2.46}$$

onde $\mathbf{b} = [b_0\, b_1\, \ldots\, b_M]^T \in \mathcal{C}^{(M+1)\times 1}$ é o vetor formado pelos coeficientes do polinômio $b(z)$ definido na Expressão (2.41).

Deste modo, o estimador ML dos coeficientes do polinômio $b(z)$, definidos pelo vetor $\mathbf{b}$, é dado por

$$\hat{\mathbf{b}}_{ML} = \arg\min_{\mathbf{b}} \mathrm{Tr}\{\mathbf{B}(\mathbf{B}^H\mathbf{B})^{-1}\mathbf{B}^H\hat{\mathbf{R}}_{\mathbf{y}}\}. \tag{2.47}$$

Uma vez determinado o vetor $\mathbf{b}$, obtemos as estimativas ML para as freqüências $\boldsymbol{\omega}$ calculando as raízes do polinômio $b(z)$ correspondente.

Restrição ao subespaço de sinal

A restrição ao subespaço de sinal é realizada através da decomposição em autovalores e autovetores de $\hat{\mathbf{R}}_{\mathbf{y}}$, dada por

$$\hat{\mathbf{R}}_{\mathbf{y}} = \ \hat{\mathbf{U}}_s\ \hat{\mathbf{\Lambda}}_s\ \hat{\mathbf{U}}_s^H + \ \hat{\mathbf{U}}_n\ \hat{\mathbf{\Lambda}}_n\ \hat{\mathbf{U}}_n^H, \tag{2.48}$$

onde $\hat{\mathbf{\Lambda}}_s$ é uma matriz diagonal $\bar{M}\times\bar{M}$ contendo os $\bar{M} = \min\{M, \mathrm{posto}(\mathbf{C})\}$ maiores autovalores de $\hat{\mathbf{R}}_{\mathbf{y}}$, os quais estão associados ao subespaço de sinal. Os $(N - \bar{M})$ autovalores restantes, referentes ao subespaço de ruído, estão na matriz diagonal $\hat{\mathbf{\Lambda}}_n$. As colunas da matriz $\hat{\mathbf{U}}_s$ $(N \times \bar{M})$ correspondem aos autovetores do subespaço de sinal, enquanto as colunas de $\hat{\mathbf{U}}_n$

$(N \times (N - \bar{M}))$ são os autovetores do subespaço de ruído.

A priori, o problema de minimização que o algoritmo MODE se propõe a resolver é dado por

$$\mathbf{b}_{MODE} = \arg \min_{\mathbf{b}} \mathrm{Tr}\{\mathbf{B}(\mathbf{B}^H\mathbf{B})^{-1}\mathbf{B}^H\hat{\mathbf{U}}_s\mathbf{\Lambda}_M\hat{\mathbf{U}}_s^H\}, \tag{2.49}$$

onde $\mathbf{b} = [b_0\, b_1\, \ldots\, b_M]^T$, $\mathbf{B}^H \in \mathcal{C}^{(N-M)\times N}$ é definida pela Expressão 2.44, $\mathbf{\Lambda}_M = (\hat{\mathbf{\Lambda}}_s - \hat{\sigma^2}\mathbf{I})^2\hat{\mathbf{\Lambda}}_s^{-1}$ e $\hat{\sigma^2} = \mathrm{Tr}\{\hat{\mathbf{\Lambda}}_n\}/(N - \bar{M})$.

É interessante compararmos a Expressão (2.49) com a Expressão (2.37), definida pelo estimador ML. As diferenças existentes estão associadas ao uso da nova parametrização, conforme detalhado na Seção 2.4.1, e ao emprego da decomposição em autovalores e autovetores da matriz de subespaço de sinal, no lugar de usar a matriz $\hat{\mathbf{R}}_{\mathbf{y}}$, incluindo uma correção nos autovalores do subespaço de sinal, na tentativa de remover a parcela de ruído.

Contudo, em vez de resolver o problema definido em (2.49), o algoritmo MODE explora a forma quadrática que a função $J_{ML}(\mathbf{b})$ pode assumir (Stoica e Sharman, 1990), e resolve o seguinte problema quadrático equivalente:

$$\boldsymbol{\beta}_{MODE} = \arg \min_{\boldsymbol{\beta}} \|\mathbf{V}\boldsymbol{\beta}\|^2, \tag{2.50}$$

onde $\|\cdot\|$ denota a norma Euclidiana, $\boldsymbol{\beta} \in \mathcal{R}^{M+1\times 1}$ é um vetor que está relacionado com os coeficientes do polinômio $b(z)$ segundo

$$\mathbf{b} = \mathbf{W}\boldsymbol{\beta}, \tag{2.51}$$

sendo que a matriz $\mathbf{W} \in \mathcal{C}^{(M+1)\times(M+1)}$ define restrições de simetria complexa conjugada sobre $\mathbf{b}$ (Krummenauer, 2007). Em (Stoica e Sharman, 1990) (Krummenauer, 2007), temos os detalhes de como obter o problema definido na Expressão (2.50).

Sob esta formulação, obtemos um problema de otimização não-linear cuja matriz $\mathbf{V}$ depende de $\mathbf{B}^H\mathbf{B}$, a qual, por sua vez, depende de $\boldsymbol{\beta}$. O método MODE propõe os seguintes passos para

resolver este problema (Stoica e Sharman, 1990):

1. resolva o problema da Expressão (2.50) usando a matriz $\mathbf{V}$ correspondente a $\mathbf{B}^H\mathbf{B} = \mathbf{I}$, de modo a determinar um valor inicial para $\boldsymbol{\beta}$.

2. calcule $\mathbf{B}^H\mathbf{B}$ usando o vetor $\boldsymbol{\beta}$ determinado no passo anterior e $\mathbf{b} = \mathbf{W}\boldsymbol{\beta}$. Então, resolva novamente o problema da Expressão (2.50), agora usando $\mathbf{V}$ atualizado com o novo valor de $\mathbf{B}^H\mathbf{B}$.

O segundo passo pode ser repetido por algumas iterações com a finalidade de melhorar a precisão das estimativas, particularmente quando o número de *snapshots* considerados for pequeno. Ao final, o vetor $\boldsymbol{\beta}$ resultante é utilizado na Expressão (2.51) para obtermos então o vetor $\mathbf{b}_{MODE}$. Por fim, as freqüências ω_m são determinadas por meio das raízes do polinômio $b(z)$ correspondente.

2.4.2 O Algoritmo MODEX

Em 1999, Alex B. Gershman e Petre Stoica (A. Gershman e Stoica, 1999) apresentaram o algoritmo MODEX (MODE *with eXtra Roots*) como uma proposta de melhoria do método MODE, sem perda da eficiência assintótica e da moderada complexidade verificadas para aquele algoritmo.

A idéia principal do algoritmo MODEX é eliminar as estimativas DOA (raízes do polinômio $b(z)$), denominadas *outliers*, que estão muito distantes dos valores verdadeiros de ω_m, $m = 1, \ldots, M$. Estes *outliers* surgem devido a discrepâncias entre os subespaços de sinal amostrado, isto é, aquele obtido usando a matriz $\hat{\mathbf{R}}_{\mathbf{y}}$, definida na Expressão (2.16), e exato, associado à matriz $\mathbf{R}_{\mathbf{y}}$, dada pela Expressão (2.15). A fim de atenuar o efeito dos *outliers*, este método emprega o vetor estendido

$$\bar{\mathbf{b}} = \begin{bmatrix} b_0 & \ldots & b_M & b_{(M+1)} & \ldots & b_{(M+P)} \end{bmatrix}^T \tag{2.52}$$

na matriz definida pela Expressão (2.44), onde P é um inteiro arbitrário pertencente ao intervalo $(0, N-M)$. O objetivo é gerar raízes extras visando melhorar os modelos para ambos subespaços de sinal e ruído.

Deste modo, o problema de minimização que o algoritmo MODEX procura resolver é equivalente àquele apresentado na Seção 2.4.1 para o MODE, exceto pela dimensão do vetor $\bar{\mathbf{b}}$. Ou seja,

$$\mathbf{b}_{MODEX} = \arg \min_{\bar{\mathbf{b}}} \mathrm{Tr}\{\bar{\mathbf{B}}(\bar{\mathbf{B}}^H\bar{\mathbf{B}})^{-1}\bar{\mathbf{B}}^H\hat{\mathbf{U}}_s\mathbf{\Lambda}_M\hat{\mathbf{U}}_s^H\} \tag{2.53}$$

onde

$$\bar{\mathbf{B}}^H = \begin{bmatrix} b_{M+P} & \cdots & b_0 & \mathbf{0} \\ & \ddots & & \ddots \\ \mathbf{0} & & b_{M+P} & \cdots & b_0 \end{bmatrix}, \tag{2.54}$$

$\bar{\mathbf{B}}^H \in \mathcal{C}^{(N-M-P)\times N}$. $\mathbf{\Lambda}_M$ e $\hat{\sigma}^2$ são definidos da mesma forma que na Seção 2.4.1.

A solução do problema definido na Expressão (2.53) pode ser alcançada por meio dos mesmos passos apresentados na Seção 2.4.1. Ao final, o vetor $\bar{\mathbf{b}}$ resultante determina as $(M+P)$ novas raízes. Entretanto, com a finalidade de preservar o desempenho assintótico que o MODE já possui, as M raízes obtidas com o MODE convencional são acrescentadas ao conjunto de raízes obtidas a partir da Expressão (2.53). Com isto, temos um conjunto de $(2M+P)$ soluções candidatas.

A seleção das M melhores raízes dentre todas as candidatas é realizada da seguinte forma: 1) todas as possíveis combinações de M raízes são geradas e os respectivos valores da função custo $J_{ML}(\boldsymbol{\theta})$ são calculados; 2) a combinação cujo valor de $J_{ML}(\boldsymbol{\theta})$ for o menor determina as estimativas MODEX para os ângulos de chegada.

Por fim, é pertinente destacarmos que a introdução de P raízes extras reduz a capacidade do MODEX, isto é, o número máximo de ângulos a serem estimados. Enquanto o MODE pode

estimar até $N-1$ ângulos de chegada, o MODEX pode estimar

$$M_{max} = N - P - 1, \tag{2.55}$$

onde $0 < P < N - M$. A restrição imposta pela Expressão (2.55) pode inviabilizar a aplicação do MODEX quando o número de sensores N é pequeno.

2.4.3 Discussão

A Figura 2.5 apresenta o desempenho dos algoritmos MODE e MODEX, junto com o limite de Cramér-Rao e a curva RMSE obtida para a busca em grade, considerando o cenário DOA descrito na Seção 2.3. Novamente, para cada valor de SNR, foram realizados $N_e = 1000$ experimentos. Além disso, foram concedidas aos algoritmos MODE e MODEX até 3 iterações para a resolução do segundo passo do procedimento apresentado na Seção 2.4.1.

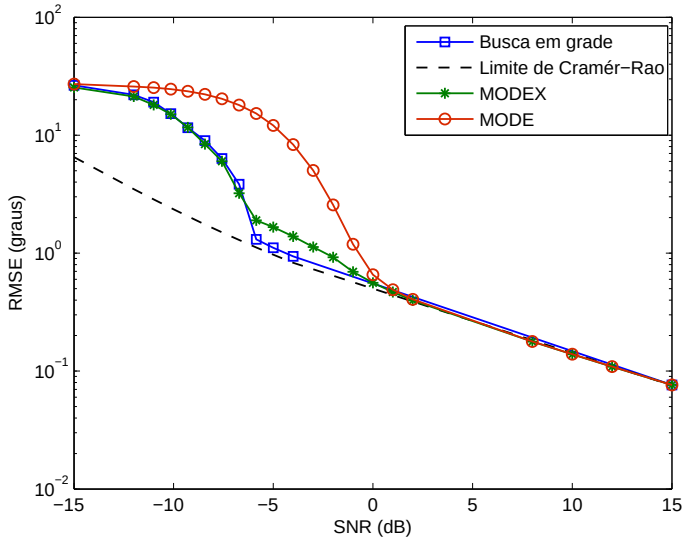


Figura 2.5: Desempenho dos algoritmos MODE e MODEX em função da SNR.

Primeiramente, podemos observar que tanto o MODE quanto o MODEX tendem assintoticamente ao limite de Cramér-Rao. Além disso, conforme a SNR aumenta, ambos conseguem atingir o desempenho ML, isto é, obtêm os mesmos valores RMSE que a busca em grade. Por

outro lado, sofrem uma degradação de desempenho à medida que a SNR é reduzida. Note que mesmo o MODEX não é capaz de reproduzir o desempenho ML perfeitamente: na região em que a SNR assume valores aproximadamente entre -6 dB e 0 dB, o MODEX comete erros de estimação (medidos pela RMSE) superiores aos obtidos pela busca em grade. Entretanto, é possível notar o ganho de desempenho em relação ao MODE que a introdução de raízes extras conferiu ao MODEX.

2.5 Conclusão

Neste capítulo, apresentamos os princípios referentes ao problema de estimação DOA. O modelo de sinal e as principais suposições estatísticas foram detalhados e serviram de base para a dedução do estimador ML para os ângulos de chegada. Como mostrado na Seção 2.2, as estimativas ML são obtidas por meio da minimização da função custo $J_{ML}(\boldsymbol{\theta})$, a qual possui um caráter não-linear, não-quadrático e multimodal. Os desafios inerentes à otimização da função $J_{ML}(\boldsymbol{\theta})$ foram discutidos e ilustrados na Seção 2.3, com destaque para o fato de que emergem múltiplos problemas de otimização sob um mesmo cenário DOA.

Por causa destas características, somente uma abordagem de busca exaustiva consegue garantidamente localizar o mínimo global de $J_{ML}(\boldsymbol{\theta})$. Contudo, em vista de seu elevado custo computacional, sua aplicação é inviável em um contexto prático geral. Diversos métodos baseados no critério ML foram propostos como alternativas à busca exaustiva. Dois deles, a saber, MODE e MODEX, foram descritos sucintamente na Seção 2.4. O desempenho destes métodos foi analisado considerando um cenário padrão na literatura DOA. Apesar de assintoticamente serem capazes de reproduzir o desempenho do estimador ML, ambos perdem qualidade na estimação conforme a SNR diminui, afastando-se assim do desempenho desejado.

Este trabalho propõe o uso de meta-heurísticas populacionais, mais especificamente, algoritmos de computação natural, para realizar a minimização da função custo $J_{ML}(\boldsymbol{\theta})$. Nosso objetivo é avaliar se estas ferramentas são capazes de atingir o desempenho do estimador ML,

mas com um custo computacional inferior. No próximo capítulo, apresentaremos os algoritmos de computação natural que serão empregados no problema de estimação de direção de chegada.

Capítulo 3

Computação Natural

3.1 Introdução

A relação entre o ser humano e a natureza é algo extraordinário. Desde os seus primórdios, esta magnífica e dinâmica interação tem colaborado para o desenvolvimento do ser humano de diversas maneiras. No início, o homem buscou na natureza os meios mais básicos para sua sobrevivência, como alimento e abrigo. Posteriormente, passou não apenas a usufruir daquilo que a natureza dispunha, mas também a agir sobre a mesma: agricultura e pecuária são exemplos de como o homem a modifica e manipula.

Além disto, o homem se dedicou à observação e ao estudo dos fenômenos e processos naturais com o propósito de melhor compreender e explicar como a natureza funciona. E, à medida que progressos neste sentido foram atingidos, novas invenções e formas de interagir com a natureza se tornaram possíveis. Por exemplo, após o aprendizado e a elaboração das leis físicas de movimento e gravidade, tornou-se possível o projeto e surgimento de aeronaves.

Com o advento dos computadores digitais, a relação homem-natureza ganhou novas perspectivas. Agora, a natureza tem servido como fonte de inspiração para a elaboração e desenvolvimento de técnicas para a resolução de problemas complexos em diversos domínios. Outra novidade é a possibilidade de simular e emular modelos e processos naturais neste ambiente

computacional, o que favorece a ampliação de nossa compreensão acerca destes fenômenos, e nos habilita a até mesmo estudar e "experimentar" formas, comportamentos e organismos inexistentes na natureza. Além disto, a investigação sobre novos materiais e meios pelos quais se pode realizar computação tem atraído atenção e também constitui uma destas perspectivas.

Computação natural é a terminologia introduzida para englobar as três abordagens mencionadas acima. Ou seja, computação natural pode ser definida como o campo de pesquisa que, baseado ou inspirado na natureza, busca o desenvolvimento de novas ferramentas computacionais (*software* ou *hardware*) para a solução de problemas, conduz à síntese e ao estudo de fenômenos, comportamentos e organismos naturais, e que almeja o projeto de novos sistemas utilizando diferentes materiais naturais para computar (de Castro, 2006).

O escopo de aplicação dos algoritmos de computação natural [1] é bastante vasto e diversificado. A seguir, destacamos algumas possíveis áreas de aplicação destas ferramentas (de Castro, 2006) (Bäck, Fogel, e Michalewicz, 2000a):

- planejamento: roteamento, *scheduling*;
- aprendizado de máquina e reconhecimento de padrões;
- análise de dados: clusterização e mineração;
- navegação autônoma e controle;
- problemas de classificação e aproximação de funções;
- otimização: multimodal, multiobjetivo, dinâmica;

No contexto deste trabalho, estamos interessados no uso de algoritmos de computação natural para a solução de um problema de otimização: encontrar o mínimo global de $J_{ML}(\boldsymbol{\theta})$, de modo a determinar a melhor estimativa dos ângulos de chegada segundo o critério ML, conforme a Expressão (2.37).

[1] Ao longo desta dissertação, a terminologia algoritmo de computação natural será utilizada como referência a ferramentas computacionais inspiradas na natureza cuja finalidade é a solução de problemas, dentre eles, problemas de otimização de funções.

3.2 Motivação

Conforme enfatizado nas Seções 2.3 e 2.4, o estimador ML para os ângulos de chegada envolve a minimização da função custo $J_{ML}(\boldsymbol{\theta})$, a qual é não-linear e não-quadrática em relação ao parâmetro $\boldsymbol{\theta}$ e possui diversos mínimos locais. Além disto, a superfície desta função varia consideravelmente conforme o valor da relação sinal-ruído é modificado.

Estas características dificultam sobremaneira a tarefa de encontrar o ponto de mínimo global, de modo que, *a priori*, somente uma abordagem de busca exaustiva é capaz de garantidamente atingir o desempenho ML. Porém, esta abordagem realiza a otimização de $J_{ML}(\boldsymbol{\theta})$ às custas de um esforço computacional demasiadamente elevado, e, portanto, não pode ser empregada. Mesmo os métodos MODE e MODEX, discutidos na Seção 2.4, falham em determinar o mínimo global para um conjunto de valores de SNR, conforme já evidenciado na literatura.

É justamente em casos como este que meta-heurísticas populacionais, entre elas, os algoritmos de computação natural, se credenciam como alternativas promissoras. Apesar de não garantirem a obtenção da solução ótima, estas ferramentas têm obtido êxito e se mostrado robustas quando lidam com a tarefa de localizar máximos ou mínimos de funções multimodais, inclusive no contexto de processamento de sinais (Attux et al., 2003). Outra característica particularmente interessante no contexto de otimização multimodal é que estas ferramentas, uma vez que utilizam uma população de soluções candidatas, tendem a realizar a exploração de diferentes regiões do espaço de busca de modo simultâneo, além de promover um aprimoramento do processo de busca pelo ótimo através de mecanismos de interação ou comunicação entre as soluções candidatas.

Segundo o Teorema *No-Free-Lunch* (Wolpert e Macready, 1995) (Wolpert e Macready, 1997), não há uma técnica de otimização capaz de resolver todos os problemas com desempenho, na média, superior a qualquer outra técnica. Em outras palavras, se o algoritmo A_1 é capaz de resolver uma determinada classe de problemas e obtém um desempenho superior ao do algoritmo A_2, é certo que existe uma outra classe de problemas para a qual a situação se inverte.

À luz deste teorema, algumas considerações acerca da aplicação de algoritmos de computação natural ao problema de estimação DOA podem ser feitas: por um lado, concluímos que, apesar de terem obtido sucesso em problemas semelhantes, não há garantias que estas ferramentas irão obter o desempenho desejado neste problema. Não obstante, é possível que este problema em particular pertença exatamente à classe de problemas de otimização na qual estas ferramentas superam as técnicas clássicas. Por estas razões, julgamos necessário, e certamente benéfico, o estudo da aplicação de computação natural ao problema de estimação DOA.

Contudo, quais são os algoritmos de computação natural mais indicados para o problema de estimação DOA? Existem critérios adequados para auxiliar e dirigir este processo de seleção? A partir do Teorema *No-Free-Lunch*, sabemos que o desempenho de um algoritmo em problemas de otimização com características semelhantes ao problema de estimação DOA certamente é um indicativo útil, porém não definitivo, para sua escolha. Outro critério pertinente está associado à robustez e ao grau de especialização de um algoritmo. Também pode ser interessante considerar o tempo e o nível de exposição de um algoritmo à comunidade científica, uma vez que aqueles já submetidos à maior análise tendem a oferecer uma avaliação mais apurada, inclusive teórica, acerca de seu desempenho. Entretanto, não existe um consenso no que se refere à escolha de algoritmos, tampouco sobre os critérios ideais para fazê-la.

Considerando os aspectos apontados anteriormente, foram selecionados algoritmos vinculados a três linhas de pesquisa pertencentes à computação natural, a saber: computação evolutiva, sistemas imunológicos artificiais e inteligência de enxame. Em termos simples, a computação evolutiva provê ferramentas computacionais para a solução de problemas, incluindo otimização, tendo como inspiração os princípios da moderna teoria da evolução (Neo-Darwinismo) (Bäck et al., 2000a) (de Castro, 2006). Em contrapartida, a classe dos sistemas imunológicos artificiais está vinculada a elementos conceituais derivados de teorias que versam sobre o funcionamento do sistema de defesa dos mamíferos (Castro e Timmis, 2002), ao passo que as técnicas de otimização baseadas em inteligência de enxame surgiram a partir da idéia de reproduzir em computador a forma pela qual as sociedades coletivas processam o conhecimento (Kennedy e

Eberhart, 1995) (Kennedy, 1997).

As seções a seguir contêm a descrição detalhada dos algoritmos de computação natural utilizados neste trabalho.

3.3 Prólogo

Antes de começarmos a descrever os princípios de cada algoritmo de computação natural, é necessário definir três conceitos básicos: a representação, o objetivo e a função de avaliação (*fitness*, em inglês) (Michalewicz e Fogel, 2004). A representação determina como cada solução candidata para um determinado problema está codificada, e, conseqüentemente, como pode ser manipulada. O objetivo estabelece a meta a ser atingida, isto é, qual o problema que se deseja resolver, enquanto a função de avaliação fornece uma medida da qualidade de cada solução candidata dada sua representação, permitindo assim uma comparação entre soluções alternativas.

No caso específico do problema de estimação DOA, uma vez que estamos interessados em determinar o vetor $\boldsymbol{\phi} = [\phi_1 \ldots \phi_M]^T$ dos ângulos de chegada, é conveniente representarmos cada solução candidata como um vetor de parâmetros reais. O objetivo, por sua vez, está dado na Expressão (2.37): minimizar a função custo $J_{ML}(\boldsymbol{\theta})$, de modo a encontrar as melhores estimativas dos ângulos de chegada segundo o critério ML.

Precisamos, por fim, definir uma função de avaliação $F_{fit}(\mathbf{x}) : \mathcal{R}^M \longmapsto \mathcal{R}$ para o problema de estimação DOA. Dado um vetor $\mathbf{x} \in \mathcal{R}^M$ com estimativas para os ângulos de chegada, a função $F_{fit}(\mathbf{x})$ deve retornar um valor indicando a qualidade destas estimativas.

A própria função $J_{ML}(\boldsymbol{\theta})$ realiza este mapeamento e, portanto, poderia ser utilizada como função de avaliação. Entretanto, uma vez que os algoritmos de computação natural usualmente operam no sentido de privilegiar as soluções com maior *fitness*, é conveniente convertermos o problema de minimização de $J_{ML}(\boldsymbol{\theta})$ em um problema de maximização equivalente. Dentre as diversas formas de realizar esta conversão, definimos uma transformação monotônica e

inversamente proporcional ao valor de $J_{ML}(\boldsymbol{\theta})$, expressa por

$$F_{fit}(\boldsymbol{\theta}) = \frac{1}{1 + J_{ML}(\boldsymbol{\theta})}. \tag{3.1}$$

Desta maneira, os algoritmos de computação natural buscarão localizar o máximo global da função de avaliação $F_{fit}(\boldsymbol{\theta})$, o qual corresponde exatamente ao mínimo global de $J_{ML}(\boldsymbol{\theta})$, que, por sua vez, segundo o critério ML, determina as melhores estimativas para os ângulos de chegada.

Definidos estes três conceitos, passamos à descrição dos algoritmos de computação natural que foram utilizados neste trabalho.

3.4 Computação Evolutiva

Em 1859, Darwin propôs que uma população de indivíduos capazes de se reproduzir, sujeita à variação genética seguida por um processo de seleção, tende a resultar em novas populações cada vez melhor adaptadas ao ambiente (Darwin, 1859). Esta teoria serviu de inspiração para o surgimento de métodos computacionais, denominados algoritmos evolutivos, para a solução de problemas em diversos contextos, desde controle de processos até otimização. Computação Evolutiva é o termo utilizado para descrever o campo de pesquisa que engloba todos os algoritmos evolutivos, sendo um dos principais ramos dentro da Computação Natural.

A idéia básica da computação evolutiva é, portanto, fazer uso do processo de evolução natural como um paradigma para a solução de problemas (Bäck et al., 2000a). As principais frentes de pesquisa em computação evolutiva são: os algoritmos genéticos, as estratégias evolutivas, a programação evolutiva e a programação genética (Bäck et al., 2000a) (Bäck, Fogel, e Michalewicz, 2000b). A despeito de suas diferenças, todas estas abordagens utilizam os princípios básicos do processo evolutivo como proposto pela teoria da evolução de Darwin e pela genética moderna. Abaixo, apresentamos a estrutura básica de um algoritmo evolutivo (Bäck

et al., 2000a) (de Castro, 2006):

a) reprodução com herança: cada indivíduo representa ou codifica uma solução potencial para o problema. Estes indivíduos podem se reproduzir (de forma sexuada ou assexuada), gerando descendentes que herdam algumas características de seu(s) pai(s).

b) variação genética: os descendentes são submetidos a variações genéticas através de mutação. Este processo de mutação permite a aparição de novas características genéticas na população e, portanto, a exploração de novas regiões do espaço de busca associado ao problema.

c) seleção natural: os indivíduos presentes na população competem pela sobrevivência e pelo direito de se reproduzir. O processo de seleção utiliza a medida de adaptação (*fitness*) de cada indivíduo como principal critério de escolha, privilegiando aqueles indivíduos que apresentam maior valor de *fitness*.

Os algoritmos evolutivos que utilizamos neste trabalho, embora possuam características particulares, empregam as três operações fundamentais - recombinação ou *crossover*, mutação e seleção - para manipular a população de soluções candidatas e explorar o espaço de busca. Contudo, existe uma grande diversidade de operadores propostos para realizar cada operação. Nas seções a seguir, descrevemos detalhadamente os algoritmos evolutivos utilizados, bem como os operadores escolhidos para cada um deles.

3.4.1 Algoritmo Genético

Os algoritmos genéticos (em inglês, *genetic algorithms* (GAs)) foram propostos por J. Holland em 1975 (Holland, 1975), embora tenham suas raízes em trabalhos mais antigos, como (Friedberg, 1958), (Fraser, 1959) e (Bremermann, 1962). Estes algoritmos são caracterizados pela uso da estrutura básica apresentada na Seção 3.4.

Em primeiro lugar, o algoritmo gera aleatoriamente uma população inicial de N_P indivíduos, isto é, a matriz **Pop** com N_P vetores solução de dimensão M (cromossomos), onde M é o número de ângulos de chegada. Uma vez que os ângulos de chegada ϕ_i, $i = 1, \ldots, M$, assumem valores no intervalo range $=(-90°, 90°)$, cada componente Pop_{ij} é dada, portanto, por uma amostra de uma variável aleatória com distribuição uniforme neste intervalo. Inicializada a população, a medida de *fitness* de cada indivíduo é determinada e armazenada em **fPop**.

Em seguida, o algoritmo entra no laço principal, no qual a população é submetida aos operadores genéticos de recombinação, mutação e seleção, respectivamente.

Operador de Crossover

A etapa de recombinação começa com a seleção dos indivíduos que irão se reproduzir e a formação dos pares. O parâmetro p_c, denominado probabilidade de *crossover*, define a probabilidade de um indivíduo ser selecionado para este fim. Este parâmetro é único, e, portanto, todos os indivíduos presentes na população possuem as mesmas chances de se reproduzirem, independentemente do *fitness*.

Em cada iteração, a partir da população de N_P indivíduos, N_f indivíduos são selecionados para reprodução por meio do seguinte procedimento:

- Para cada indivíduo $\mathbf{Pop_i}$: se $u_x \leq p_c$, onde $u_x \backsim \mathcal{U}(0, 1)$, então o indivíduo $\mathbf{Pop_i}$ é selecionado.

Segue-se então para a formação dos pares: para cada indivíduo dentre os N_f escolhidos, um outro indivíduo pertencente a este mesmo conjunto é selecionado aleatoriamente (sem reposição). Procede-se então à geração dos descendentes.

Empregamos neste trabalho o operador denominado *crossover* aritmético (Michalewicz, 1996) (Bäck et al., 2000a). Segundo este operador, os descendentes são formados a partir de combinações lineares dos indivíduos originais. Sejam, portanto, $\mathbf{x} = [x_1 \ldots x_M]^T$ e $\mathbf{y} = [y_1 \ldots y_M]^T$ dois indivíduos selecionados para reprodução. As componentes dos novos

indivíduos, $\bar{\mathbf{x}} = [\bar{x}_1 \ \ldots \bar{x}_M]^T$ e $\bar{\mathbf{y}} = [\bar{y}_1 \ \ldots \bar{y}_M]^T$, são obtidas conforme as expressões:

$$\bar{x}_i = \alpha x_i + (1 - \alpha) y_i \tag{3.2}$$

$$\bar{y}_i = (1 - \alpha) x_i + \alpha y_i, \tag{3.3}$$

onde $\alpha \backsim \mathcal{U}(0, 1)$.

Uma vez gerados os descendentes, estes são acrescentados à população original, formando a população intermediária **Temp**, a qual é submetida ao processo de mutação.

Operador de Mutação

Usualmente, os operadores de mutação geram novos indivíduos (descendentes) a partir de um único indivíduo original. Considerando um vetor real $\mathbf{r} = [r_1 \ \ldots r_M]^T$, uma forma bastante comum de gerar um descendente $\bar{\mathbf{r}} = [\bar{r}_1 \ \ldots \bar{r}_M]^T$ por mutação é dada por:

$$\bar{r}_i = r_i + m_r, \tag{3.4}$$

onde i denota à i-ésima componente de $\mathbf{r}$ que foi selecionada para mutação e m_r é uma variável aleatória. As componentes que devem sofrer mutação são determinadas por meio de um procedimento similar à seleção de indivíduos para *crossover*, conforme descrito anteriormente, com a diferença de agora utilizar o parâmetro p_m, denominado probabilidade de mutação, o qual define a probabilidade de uma componente (gene) de um indivíduo sofrer mutação.

Desde os trabalhos de (Rechenberg, 1973) e (Schwefel, 1981), o uso de uma variável aleatória com distribuição normal tem sido uma das principais maneiras de introduzir mutação em vetores de parâmetros reais. Neste sentido, é conveniente empregar uma variável aleatória gaussiana de média zero e desvio padrão σ_m para efetuar a mutação por três motivos: 1) na média, os descendentes gerados não são diferentes dos indivíduos originais, isto é, $E\{\bar{r}\} = E\{r\}$. 2) é cada vez menos provável que os descendentes sejam muito diferentes dos indivíduos originais.

3) o parâmetro σ_m oferece a possibilidade de regular o passo (tamanho) da mutação (Bäck et al., 2000a). Por estas razões, este foi o operador de mutação empregado neste trabalho.

Após sofrer mutação, a população é novamente avaliada. Contudo, somente os indivíduos que sofreram mutação e os indivíduos gerados na etapa de *crossover* é que efetivamente precisam ser avaliados. Por isso, com a finalidade de reduzirmos o número total de avaliações da função de *fitness*, associamos a cada indivíduo um bit de sinalização: o valor 0 indica que o indivíduo não precisa ser avaliado, enquanto o valor 1 indica que o mesmo ainda não foi avaliado (indivíduo gerado via *crossover*), ou que a medida de *fitness* existente não está atualizada (o indivíduo sofreu mutação) (Bäck et al., 2000b).

Por fim, tendo a população intermediária **Temp** e os valores de *fitness* de cada indivíduo (vetor **fTemp**), o algoritmo então seleciona os N_P que passarão para a próxima iteração (geração).

Operador de Seleção

A função precípua do operador de seleção é conduzir a busca pelo ótimo global através da escolha das soluções candidatas que permanecerão na população. Esta escolha deve ser feita no sentido de privilegiar as soluções de maior *fitness*, para que as regiões promissoras do espaço de busca já identificadas sejam melhor exploradas. Contudo, é essencial que o operador de seleção também preserve diversidade na população, de modo a ampliar as chances de novas regiões do espaço de busca serem exploradas.

Dentre os diversos operadores de seleção propostos na literatura, escolhemos para este trabalho o método de seleção por torneio (em inglês, *tournament selection*). Em suma, este operador executa a seleção da seguinte maneira:

1. A partir de **Temp**, são extraídos N_t indivíduos (com ou sem reposição), formando o conjunto P_t com os indivíduos que irão competir em um torneio.

2. O vencedor do torneio será o indivíduo de P_t com o maior valor de *fitness*. Este

indivíduo é, então, inserido na população **Pop** que passará para a próxima iteração.

O procedimento descrito anteriormente é repetido N_P vezes para que se complete a população **Pop**. É interessante notarmos que, dependendo do valor definido para o parâmetro N_t, a pressão seletiva que este operador exerce sobre a população pode variar: quanto maior o valor de N_t, mais indivíduos participarão de cada torneio, aumentando assim as chances de indivíduos com valores elevados de *fitness* serem escolhidos mais vezes, o que caracteriza uma forte pressão seletiva. De maneira análoga, quanto menor o valor de N_t, menor será a pressão seletiva.

Esta flexibilidade é bastante conveniente no contexto da aplicação do algoritmo genético ao problema de estimação DOA, uma vez que estaremos interessados em manter diversidade na população na tentativa de escapar das soluções sub-ótimas existentes, principalmente nas situações discutidas na Seção 2.3.

Também é interessante notarmos que este operador permite que o melhor indivíduo da população seja eliminado da população na iteração seguinte: basta que o mesmo não participe de nenhum torneio. Para evitarmos isto, utilizamos uma abordagem elitista em conjunto com a seleção: após a realização dos N_P torneios, caso o melhor indivíduo não esteja em **Pop**, ele é reinserido, substituindo o indivíduo com menor valor de *fitness*.

Feita a seleção, o algoritmo genético conclui uma iteração (geração). Este processo iterativo é repetido até que algum critério de parada seja atingido. Neste trabalho, por conveniência, optamos pelo uso de um número máximo de iterações ($\max_{it}$) como o critério de parada de todos os algoritmos de computação natural. O Quadro 1 apresenta o pseudo-código do algoritmo genético aplicado ao problema de estimação DOA.

Quadro 1 Algoritmo Genético

Função [**Pop**] = GA(N_P,M,range,p_m,p_c,N_t,F_{fit})
[**Pop**, **fPop**] ⇐ inicializa(N_P,M,range,F_{fit})
Enquanto critério de parada não for satisfeito **faça**
  Off ⇐ crossover(**Pop**, p_c)
  Temp ⇐ [**Pop**, **Off**]
  Temp ⇐ mutação(**Temp**, p_m)
  fTemp ⇐ avalie(**Temp**,F_{fit})
  [**Pop**, **fPop**] ⇐ seleção(**Temp**, **fTemp**, N_P, N_t)
Fim Enquanto

Número de avaliações da função de fitness

Para completarmos a descrição do algoritmo genético empregado neste trabalho, derivamos a seguir a expressão para o número médio de avaliações da função de *fitness*. Esta métrica, a qual fornece um indicativo de quantos pontos do espaço de busca foram amostrados e utilizados pelo algoritmo, será útil para a análise do desempenho dos algoritmos de computação natural a ser feita nos Capítulos 4 e 5.

1. População inicial: são realizadas N_P avaliações.

2. Em cada uma das $\max_{it}$ iterações, somente os indivíduos gerados via *crossover* e aqueles que sofrem mutação é que são avaliados.

- Na média, $(p_c \cdot N_P)$ descendentes são gerados pelo operador de *crossover* e, portanto, precisam ser avaliados.
- Dos N_P indivíduos presentes na população **Pop**, na média, $(p_{1m} \cdot N_P)$ indivíduos sofrem mutação, onde p_{1m} corresponde à probabilidade de um indivíduo sofrer mutação. Se pelo menos uma componente de um indivíduo sofre mutação, é necessário avaliá-lo novamente. Assim, seja p_m a probabilidade de uma componente sofrer mutação e M o número de componentes de cada indivíduo. A probabilidade p_{1m} é definida pela binomial:

$$p_{1m} = \sum_{k=1}^{M} \binom{M}{k} {p_m}^k (1 - p_m)^{M-k}. \tag{3.5}$$

Portanto, na média, o número total de avaliações da função de *fitness* para o algoritmo genético é dado por:

$$N_{fit} = N_P + \max_{it}(p_c N_P + p_{1m} N_P). \tag{3.6}$$

3.4.2 Fitness Sharing

Os algoritmos géneticos clássicos são ferramentas poderosas no contexto de otimização de funções. Entretanto, experimentos e análises mostram que estas técnicas não são particularmente eficientes em localizar os múltiplos ótimos de uma superfície multimodal, uma vez que não estão adaptadas para preservar diversas soluções promissoras durante a busca (Mahfoud, 1995a).

Inspirados no conceito de nichos ecológicos, os quais representam regiões do ambiente que suportam diferentes tipos de vida compartilhando os recursos disponíveis, vários métodos foram desenvolvidos com o propósito de manter a diversidade na população e permitir a investigação de múltiplos ótimos de modo paralelo. Tais mecanismos, conhecidos como métodos de *niching*, criam e mantêm subpopulações nas vizinhanças das soluções ótimas (Mahfoud, 1995a).

Um nicho é caracterizado por uma porção limitada de recursos disponíveis para os indivíduos que nele se encontram. Cada indivíduo em um nicho tem direito a uma fração dos recursos disponíveis - quanto maior o tamanho da subpopulação, menor a fração. No contexto de otimização de funções multimodais, os recursos disponíveis em um nicho estão associados à medida de *fitness* do indivíduo. As técnicas de *niching* atuam sobre esta medida com a intenção de impedir que todos convirjam para a mesma região do espaço de busca, mantendo assim a diversidade da população.

O método de *fitness sharing* (Goldberg e Richardson, 1987) propõe a redução do *fitness* dos indivíduos da população por um fator relacionado ao número de indivíduos similares na

população. Tipicamente, o *fitness* compartilhado $\bar{f}_i$ do i-ésimo indivíduo é dado por:

$$\bar{f}_i = \frac{f_i}{nc_i}, \tag{3.7}$$

onde f_i corresponde ao *fitness* original do i-ésimo indivíduo. O fator nc_i mede o número de indivíduos com os quais o indivíduo i compartilha o *fitness* f_i, sendo calculado como uma soma de uma função de compartilhamento (em inglês, *sharing*) sobre todos os membros da população, como mostra a expressão abaixo:

$$nc_i = \sum_{k=1}^{N_P} sh(d_{ik}), \tag{3.8}$$

onde d_{ik} representa a distância entre os indivíduos i e k.

A função de compartilhamento (sh) mede o grau de similaridade entre dois indivíduos da população a partir da distância entre eles. Esta função retorna o valor 1 caso os indivíduos sejam idênticos, o valor 0, caso a distância entre eles seja superior a um limiar de similaridade, e um valor intermediário para níveis de similaridade intermediários. Uma função de *sharing* bastante empregada é dada por (Goldberg, 1989) (Sareni e Krahenbuhl, 1998):

$$sh(d_{ik}) = \begin{cases} 1 - (d_{ik}/\sigma_s)^{\alpha_s}, & \text{se } d_{ik} \leq \sigma_s \\ 0, & \text{caso contrário} \end{cases}, \tag{3.9}$$

onde σ_s denota o limiar de similaridade e α_s é uma constante que regula a forma da função de *sharing*, sendo que, usualmente, $\alpha_s = 1$ (Goldberg, 1989). Como mencionado na Seção 3.3, os indivíduos da população estão representados como vetores de parâmetros reais. Por isto, a métrica de distância empregada neste trabalho é a distância Euclidiana. O Quadro 2 apresenta o pseudo-código do método de *fitness sharing* aplicado ao problema de estimação DOA.

É importante notarmos a semelhança entre o método de *fitness sharing*, mostrado no Quadro 2, e o algoritmo genético, como apresentado na Seção 3.4.1. Conforme esperado, a única diferença está na etapa de avaliação da população: agora, o *fitness* compartilhado substitui o

Quadro 2 *Fitness Sharing*

Função [**Pop**] = fitness_sharing(N_P,M,range,p_m,p_c,σ_s,F_{fit})
[**Pop**, **fPop**] ⇐ inicializa(N_P,M,range,F_{fit})
Enquanto critério de parada não for satisfeito **faça**
  Off ⇐ crossover(**Pop**, p_c, σ_s)
  Temp ⇐ [**Pop**, **Off**]
  Temp ⇐ mutação(**Temp**, p_m)
  fTemp ⇐ avalie(**Temp**,F_{fit})
  fTemp ⇐ *sharing*(**Temp**,**fTemp**,σ_s)
  [**Pop**, **fPop**] ⇐ seleção(**Temp**, **fTemp**, N_P)
Fim Enquanto

fitness convencional.

A princípio, o método de *fitness sharing* pode utilizar os mesmos operadores que o algoritmo genético. Contudo, esta técnica preferencialmente deve trabalhar em conjunto com operadores de seleção de baixa pressão seletiva e que favoreçam a estabilidade dos nichos. Além disto, é conveniente impôr restrições durante a etapa de *crossover* com a finalidade de evitar o cruzamento entre indivíduos que pertençam a nichos diferentes (Bäck et al., 2000b) (Sareni e Krahenbuhl, 1998). Por isso, descrevemos a seguir as modificações ocorridas nas etapas de *crossover* e seleção. Enfatizamos que o mesmo operador de mutação, a saber, mutação gaussiana, foi utilizado pelo *fitness sharing*. Sua descrição pode ser encontrada na Seção 3.4.1.

Operador de Crossover

Como descrito na Seção 3.4.1, os primeiros passos durante a etapa de *crossover* consistem na seleção dos indivíduos que irão se reproduzir e na formação dos pares. À semelhança do procedimento utilizado para o algoritmo genético, a seleção é feita com base no parâmetro p_c, de modo que, novamente, todos os indivíduos têm as mesmas chances de se reproduzir.

Conforme destacado anteriormente, as técnicas de *niching* visam explorar simultaneamente múltiplos ótimos da superfície da função de *fitness*. Para isto, estas técnicas tentam criar e manter soluções candidatas representando os diferentes picos. Nesta direção, os métodos de especiação (em inglês, *speciation*) impõem restrições à formação dos pares para *crossover* à medida que promovem o cruzamento entre indivíduos similares e desfavorecem o cruzamento

entre indivíduos relacionados a picos distintos.

Duas soluções candidatas que, segundo uma métrica adequada, apresentam um grau elevado de similaridade, tendem a representar o mesmo pico da função de *fitness*. Logo, quando o cruzamento entre as mesmas é realizado, as soluções geradas, que herdam características dos pais e, por isso, são semelhantes a estes, também tendem a representar o mesmo pico. Deste modo, a imposição de uma restrição para cruzamento entre indivíduos do mesmo nicho tende a reduzir a formação de soluções letais (soluções que não estão associados a nenhum pico) e também favorece a exploração local dentro de cada nicho (Bäck et al., 2000b).

Por este motivo, o *fitness sharing* foi aplicado ao problema de estimação DOA em conjunto com um método de especiação proposto por K. Deb (Deb, 1989). Este método utiliza um esquema de restrição de cruzamento baseado na distância entre indivíduos: para determinar o par para o indivíduo $\mathbf{x_i}$, um outro indivíduo é escolhido aleatoriamente e a distância entre eles é computada. Caso seu valor seja inferior a um parâmetro σ_{mating}, então estes indivíduos participam da operação de *crossover*[2]. Caso contrário, outro indivíduo é escolhido aleatoriamente e a distância entre eles é computada. Este procedimento segue até que um indivíduo satisfaça o critério de similaridade ou que todos os membros da população tenham sido considerados. Neste último caso, um indivíduo é aleatoriamente escolhido como o par de $\mathbf{x_i}$.

Operador de Seleção

Uma vez que as técnicas de *niching* tentam realizar uma busca simultânea por múltiplos ótimos da superfície de *fitness*, é necessário que, ao longo das iterações, indivíduos representando os diferentes picos sejam preservados. Ou seja, o operador de seleção não apenas deve privilegiar as melhores soluções, mas também favorecer a formação e manutenção dos nichos.

Uma opção muito utilizada que contribui para a estabilidade dos nichos é o operador de seleção proposto por J. E. Baker (Baker, 1987), denominado *stochastic universal sampling*

[2]Uma vez que σ_{mating} e σ_s representam limiares de similaridade entre indivíduos, é comum utilizar o mesmo valor para ambos (Deb e Goldberg, 1989).

(SUS). Assim como o operador de seleção via roleta (Holland, 1975), o SUS associa à cada indivíduo uma probabilidade de seleção proporcional a seu *fitness*, conforme a expressão

$$Pr_i = \frac{f_i}{\sum_{k=1}^{N_P} f_k}, \tag{3.10}$$

onde f_i denota o *fitness* do i-ésimo indivíduo. Entretanto, este operador se mostra eficiente e apresenta uma variância menor no número de descendentes atribuídos a cada indivíduo da população, quando comparado ao operador de seleção via roleta (Baker, 1987).

Seja então **Pr** o vetor com as probabilidades de seleção de cada indivíduo, determinadas segundo a Expressão (3.10). A partir de uma população de μ_P indivíduos, o operador SUS seleciona λ_P indivíduos através do procedimento apresentado no Quadro 3. Como saída, temos o vetor $\mathbf{c} = [c_1 \ \ldots \ c_{\mu_P}]$ contendo o número de descendentes associado a cada indivíduo ($\sum c_i = \lambda_P$). Pode-se mostrar que o número esperado de descendentes que o SUS atribui ao i-ésimo indivíduo é igual a $\lambda_P Pr_i$ (Baker, 1987).

Quadro 3 *Stochastic Universal Sampling*

```
Função [c] = SUS(μ_P,λ_P,Pr)
amostre u ∽ U(0, 1/λ_P)
s ⇐ 0
Para i = 1 até μ_P faça
   c_i ⇐ 0
   s ⇐ s + Pr_i
   Enquanto u < s faça
      c_i ⇐ c_i + 1
      u ⇐ u + 1/λ_P
   Fim Enquanto
Fim Para
```

Número de avaliações da função de fitness

O cômputo do número médio de avaliações da função de *fitness* associado ao método de *fitness sharing* é apresentado a seguir:

1. População inicial: são realizadas N_P avaliações.

2. Em cada uma das $\max_{it}$ iterações, a população intermediária formada pela união dos pais com os descendentes gerados é avaliada. Como temos N_P indivíduos em **Pop**, o operador de *crossover* gera, na média, $p_c \cdot N_P$ descendentes. Logo, $(N_P + p_c \cdot N_P)$ avaliações são feitas por iteração.

Portanto, na média, o número total de avaliações da função de *fitness* para o *fitness sharing* é dado por:

$$N_{fit} = N_P + \max_{it}(p_c N_P + N_P). \tag{3.11}$$

3.4.3 Fitness Scaling

O método denominado *fitness scaling* corresponde a uma proposta para aumentar a eficiência do *fitness sharing* através de uma modificação na forma de se computar o *fitness* compartilhado. O objetivo é ressaltar os ótimos da superfície de *fitness* em relação às respectivas vizinhanças (Goldberg, 1989) (Darwen e Yao, 1995).

Uma maneira de se fazer isto envolve o uso de uma potência β_s, como mostra a expressão

$$\bar{f}_i = \frac{{f_i}^{\beta_s}}{nc_i}, \tag{3.12}$$

onde nc_i é obtido por meio da Expressão (3.8). É necessário, entretanto, realizar uma escolha adequada para o parâmetro β_s:

- se o valor de β_s for muito elevado, indivíduos situados em regiões muito próximas a um ótimo terão *fitness* muito superior aos dos outros indivíduos, o que aumenta significativamente as chances de a população rapidamente perder diversidade, o que pode comprometer a busca pelo ótimo global.

- se o valor de β_s for muito pequeno, a diferenciação entre os picos e as regiões que os cercam pode ser insuficiente, de modo que os indivíduos da população tendem a permanecer apenas nas vizinhanças dos ótimos.

Esta é a única distinção entre o método de *fitness scaling* e o *fitness sharing*. Os operadores empregados, bem como o procedimento básico realizado pelo *fitness scaling*, são equivalentes aos descritos na Seção 3.4.2, de modo que omitimos aqui sua descrição.

3.4.4 Clearing

O método de *clearing* (Pétrowski, 1996) difere do de *fitness sharing* na medida em que os recursos de um determinado nicho não são compartilhados, mas totalmente atribuídos aos melhores indivíduos. Basicamente, o método de *clearing* preserva o *fitness* dos melhores indivíduos de um nicho enquanto zera o *fitness* de todos os outros indivíduos da mesma subpopulação. O número máximo de indivíduos aceitos em cada nicho é denominado capacidade do nicho (κ).

Assim como no caso do *fitness sharing*, uma medida de similaridade entre os indivíduos é utilizada para determinar se estes pertencem à mesma subpopulação. Considera-se que dois indivíduos pertencem ao mesmo nicho se esta medida de similaridade, dada pela distância entre eles, é inferior a um limiar σ_s.

O Quadro 4 exibe o pseudo-código do método de *clearing* aplicado ao problema de estimação DOA.

Quadro 4 *Clearing*

Função [**Pop**] = clearing(N_P,M,range,p_m,p_c,σ_s,κ,F_{fit})
[**Pop**, **fPop**] ⇐ inicializa(N_P,M,range,F_{fit})
Enquanto critério de parada não for satisfeito **faça**
  Off ⇐ crossover(**Pop**, p_c, σ_s)
  Temp ⇐ [**Pop**, **Off**]
  Temp ⇐ mutação(**Temp**, p_m)
  fTemp ⇐ avalie(**Temp**,F_{fit})
  fTemp ⇐ *clearing*(**Temp**,**fTemp**,σ_s,κ)
  [**Pop**, **fPop**] ⇐ seleção(**Temp**, **fTemp**, N_P)
Fim Enquanto

Como é possível observar, a única diferença em relação ao método de *fitness sharing* é a realização da etapa de *clearing*, em vez de calcular o *fitness* compartilhado dos indivíduos da população.

Os demais operadores utilizados pelo método de *clearing* neste trabalho são equivalentes aos descritos para o *fitness sharing*. Por conseqüência, o número de avaliações da função de *fitness* também não se altera, sendo, portanto, definido pela Expressão (3.11).

3.4.5 Estratégias Evolutivas

As estratégias evolutivas (em inglês, *evolution strategies* (ES)) surgiram durante os anos 60, na Alemanha, como fruto dos trabalhos de I. Rechenberg, H.-P. Schwefel e P. Bienert (Rechenberg, 1965) (Schwefel, 1965). Inicialmente, estas técnicas foram empregadas em contextos relacionados a mecânica de fluidos, particularmente no projeto de objetos flexíveis tendo como objetivo a obtenção de uma configuração com o menor arrasto (Rechenberg, 1965). Posteriormente, foram generalizadas para lidarem com problemas de otimização de funções, com ênfase em funções de parâmetros reais (Rechenberg, 1971) (Schwefel, 1977).

Embora as estratégias evolutivas também sejam caracterizadas pelo emprego do esqueleto básico de um algoritmo evolutivo, como aquele detalhado no início da Seção 3.4, estas ferramentas apresentam uma marca distintiva: cada indivíduo possui em seu cromossomo não apenas os parâmetros associados à função de *fitness* que desejamos otimizar, isto é, o vetor de atributos, como também alguns parâmetros que controlam a distribuição da variável aleatória utilizada na mutação (Schwefel, 1981) (Beyer e Schwefel, 2002). Observe que este tipo de representação incorpora parâmetros de controle à própria codificação dos indivíduos, o que é uma idéia bastante interessante, pois permite que o processo evolutivo adapte estes parâmetros juntamente com o vetor de atributos. Este procedimento é conhecido como auto-adaptação (Bäck et al., 2000a).

Usualmente, mutações em vetores de parâmetros reais são feitas segundo uma distribuição normal multivariável de média zero e matriz de covariância $\mathbf{C_m}$ simétrica e definida positiva. Exceto quando a matriz $\mathbf{C_m}$ é diagonal, as mutações em cada direção (coordenada) apresentam correlações. G. Rudolph (Rudolph, 1992) demonstrou que uma matriz é definida positiva se, e

somente se, puder ser decomposta como $\mathbf{C_m} = (\mathbf{ST})^T(\mathbf{ST})$, onde $\mathbf{S}$ é uma matriz diagonal cujos elementos não nulos correspondem aos desvios padrões ${\sigma_m}^{(i)}, i = 1, \ldots, N_\sigma$, onde N_σ define o número de direções, e

$$\mathbf{T} = \prod_{i=1}^{N_\sigma - 1} \prod_{j=i+1}^{N_\sigma} \mathbf{R_{ij}}(\vartheta_{ij}). \tag{3.13}$$

A matriz $\mathbf{T}$ é obtida a partir do produto de $N_\sigma(N_\sigma - 1)/2$ matrizes de rotação $\mathbf{R_{ij}}$ com ângulos $\vartheta_{ij} \in (0, 2\pi]$, sendo que cada $\mathbf{R_{ij}}(\vartheta_{ij})$ corresponde a uma matriz identidade de dimensão N_σ em que quatro elementos específicos são substituídos por $r_{ii} = r_{jj} = \cos\vartheta_{ij}$ e $r_{ij} = -r_{ji} = -\sin\vartheta_{ij}$.

Como conseqüência deste resultado, podemos concluir que a definição de N_σ parâmetros de escala ${\sigma_m}^{(i)}$ e de $N_\sigma(N_\sigma - 1)/2$ ângulos é suficiente para gerar vetores aleatórios $\mathbf{z}$ com uma distribuição normal arbitrária de média zero e matriz de covariância $\mathbf{C_m}$, através da Expressão $\mathbf{z} = \mathbf{TSn_g}$, onde $\mathbf{n_g} \backsim \mathcal{N}(\mathbf{0}, \mathbf{1})$ é um vetor de componentes descorrelacionadas com distribuição normal.

No caso do algoritmo genético aplicado ao problema de estimação DOA, cada indivíduo originalmente corresponde a um vetor $\mathbf{Pop_i} \in \mathcal{R}^{M\times 1}$, onde M é o número de ângulos de chegada e $i = 1, \ldots, N_P$, sendo N_P o número de indivíduos presentes na população $\mathbf{Pop}$ (vide Seção 3.4.1). Agora, cada indivíduo será representado da seguinte forma:

$$\mathbf{Pop_i} = \begin{bmatrix} \boldsymbol{\varphi}^T & \boldsymbol{\sigma_m} & \boldsymbol{\vartheta} \end{bmatrix}^T, \tag{3.14}$$

onde $\boldsymbol{\varphi} \in \mathcal{R}^{M\times 1}$ denota as estimativas dos ângulos de chegada e corresponde ao vetor de atributos, $\boldsymbol{\sigma_m} = [{\sigma_m}^{(1)} \ldots {\sigma_m}^{(M)}]$ é o vetor que define o desvio padrão em cada direção e $\boldsymbol{\vartheta} = [\vartheta_1 \ldots \vartheta_{M(M-1)/2}]$ é o vetor com os ângulos de rotação[3].

O Quadro 5 apresenta o pseudo-código das estratégias evolutivas aplicadas ao problema de estimação de direção de chegada.

Em suma, conforme mostra o Quadro 5, a partir de uma população inicial com μ indivíduos,

[3]Para simplificar a notação e as operações, optamos pelo uso de vetores linha nesta seção

Quadro 5 Estratégia Evolutiva

Função [**Pop**] = ES(μ,λ,M,range,F_{fit})
[**Pop**, **fPop**] $\Leftarrow$ inicializa(μ,M,range,F_{fit})
Enquanto critério de parada não for satisfeito **faça**
  Para $i = 1$ até λ **faça**
    Selecione 2 indivíduos aleatoriamente
    Recombine seus ângulos, desvios padrões e vetores de atributos
    Aplique mutação sobre os ângulos, desvios padrões e vetores de atributos
    Insira os descendentes na população **Off**
  Fim Para
  Avalie a população **Off**
  Selecione os μ melhores indivíduos de **Off** ((μ, λ)-ES) ou de [**Pop Off**] ($(\mu + \lambda)$-ES)
Fim Enquanto

uma estratégia evolutiva gera λ descendentes por meio de recombinação. Em seguida, estes descendentes sofrem mutação e as respectivas medidas de *fitness* são obtidas. Por fim, temos a etapa de seleção, na qual somente os μ melhores indivíduos são preservados para a próxima geração.

A seguir, descrevemos em detalhes os novos operadores de recombinação, mutação e seleção.

Operador de Crossover

Existem diversos operadores de *crossover* que podem ser utilizados nas estratégias evolutivas. Em sua forma mais usual, um único indivíduo é gerado pelo cruzamento de dois pais selecionados aleatoriamente. Outra prática bastante comum é o emprego de operadores distintos para os parâmetros de controle e para o vetor de atributos.

Sejam $\mathbf{Pop_i} = [\boldsymbol{\varphi}^T \, \boldsymbol{\sigma_m} \, \boldsymbol{\vartheta}]^T$ e $\mathbf{Pop_j} = [\bar{\boldsymbol{\varphi}}^T \, \bar{\boldsymbol{\sigma}}_{\boldsymbol{m}} \, \bar{\boldsymbol{\vartheta}}]^T$ dois indivíduos selecionados para *crossover*. Neste trabalho, o novo indivíduo **Off** resultante do processo de *crossover* é dado por:

$$\mathbf{Off} = \left[\; (\mathbf{U}\boldsymbol{\varphi} + (\mathbf{I} - \mathbf{U})\bar{\boldsymbol{\varphi}})^T \quad \tfrac{\boldsymbol{\sigma_m} + \bar{\boldsymbol{\sigma}}_{\boldsymbol{m}}}{2} \quad \tfrac{(\boldsymbol{\vartheta} + \bar{\boldsymbol{\vartheta}})}{2} \bmod 2\pi \; \right]^T, \tag{3.15}$$

onde $a \bmod b$ denota o resto da divisão entre a e b, $\mathbf{I} \in \mathcal{R}^{M \times M}$ indica a matriz identidade e $\mathbf{U} \in \mathcal{R}^{M \times M}$ é uma matriz aleatória diagonal cujos elementos da diagonal são iguais a zero ou um com a mesma probabilidade. Os operadores empregados no *crossover* dos vetores de atributos

e dos parâmetros de controle são denominados *crossover* uniforme e *crossover* intermediário, respectivamente (Bäck et al., 2000a) (de Castro, 2006).

Operador de Mutação

A operação de mutação empregada nas estratégias evolutivas se assemelha àquela descrita na Seção 3.4.1. Contudo, as ES permitem que cada direção (componente) possua um desvio padrão próprio e também admitem a existência de correlações entre as direções. Conforme discutido anteriormente, isto é feito através da inserção dos parâmetros de controle $\sigma_m^{(i)}, i = 1, \ldots, M$, e $\vartheta_j, j = 1, \ldots, M(M-1)/2$, na própria representação de um indivíduo.

Outra característica interessante das estratégias evolutivas é que os parâmetros de controle são adaptados pelo processo evolutivo, de modo que também estão sujeitos a mutações. Em síntese, a operação de mutação gera um novo indivíduo por meio do seguinte procedimento:

1. Seja $\boldsymbol{\vartheta} \in (0, 2\pi]^{M(M-1)/2}$ o vetor com os ângulos de rotação necessários para a construção da matriz $\mathbf{T}(\boldsymbol{\vartheta})$, definida pela Expressão (3.13). Então, a operação de mutação gera os novos ângulos $\bar{\vartheta}_i$ por meio da relação

$$\bar{\vartheta}_i = (\vartheta_i + \beta_\vartheta \cdot N_\vartheta^{(i)}) \bmod 2\pi, \tag{3.16}$$

onde β_ϑ é uma constante positiva, $N_\vartheta^{(i)} \backsim \mathcal{N}(0,1)$ e $i = 1, \ldots, M(M-1)/2$.

2. Seja $\boldsymbol{\sigma_m}$ o vetor com os desvios padrões necessários para a construção da matriz diagonal $\mathbf{S}(\boldsymbol{\sigma_m})$. Então a operação de mutação gera os novos desvios $\bar{\sigma_m}^{(i)}$ segundo

$$\bar{\sigma_m}^{(i)} = \sigma_m^{(i)} \exp(\dot{\tau} \cdot Z_\tau + \tau \cdot Z_{\sigma_m}^{(i)}), \tag{3.17}$$

onde $(\dot{\tau}, \tau) \in \mathcal{R}_+^2$, $Z_{\sigma_m}^{(i)} \backsim \mathcal{N}(0,1)$ e $i = 1, \ldots, M$. $Z_\tau \backsim \mathcal{N}(0,1)$ é o mesmo para todos os i desvios padrões (Schwefel, 1995).

3. Por fim, o novo vetor de atributos $\bar{\boldsymbol{\varphi}}$ é obtido por meio da adição de um vetor aleatório com distribuição gaussiana de média zero e matriz de covariância definida por $\bar{\boldsymbol{\sigma}}_{\boldsymbol{m}}$ e $\bar{\boldsymbol{\vartheta}}$, como mostra a Expressão

$$\bar{\boldsymbol{\varphi}}_{\boldsymbol{i}} = \boldsymbol{\varphi}_{\boldsymbol{i}} + \mathbf{T}(\bar{\boldsymbol{\vartheta}})\mathbf{S}(\bar{\boldsymbol{\sigma}}_{\boldsymbol{m}})\mathcal{N}(\mathbf{0}, \mathbf{1}). \tag{3.18}$$

De acordo com (Schwefel, 1995), uma boa heurística para a escolha dos valores das constantes utilizadas na etapa de mutação é:

$$(\beta_\vartheta, \dot{\tau}, \tau) = (5\pi/180, (2M)^{\frac{1}{2}}, (4M)^{\frac{1}{4}}). \tag{3.19}$$

Além disto, é conveniente realizar a mutação na ordem apresentada, isto é, primeiro mutar os parâmetros de controle para depois aplicar a mutação sobre o vetor de atributos (Gehlhaar e Fogel, 1996).

Operador de Seleção

Neste trabalho, duas versões consagradas de ES foram utilizadas: $(\mu + \lambda)$-ES e (μ, λ)-ES (Schwefel, 1977). No primeiro caso, a notação empregada denota uma ES que gera λ descendentes a partir de uma população com μ pais e que realiza a seleção dos μ melhores indivíduos considerando $(\mu + \lambda)$ indivíduos (pais e descendentes juntos). Observe que, deste modo, os pais somente são substituídos quando os descendentes gerados os superam em termos de *fitness*.

Por outro lado, a abreviação (μ, λ)-ES denota uma ES que também gera λ descendentes a partir de uma população com μ pais, mas que seleciona os μ melhores indivíduos apenas do conjunto de λ descendentes. Como conseqüência, temos a restrição $\lambda \geq \mu$.

Número de avaliações da função de *fitness*

Seja μ o número de indivíduos presentes na população de pais e λ o número de descendentes gerados em cada iteração. O número total de avaliações da função de *fitness* realizadas por uma estratégia evolutiva é dado por:

1. População inicial: são realizadas μ avaliações.

2. Em cada uma das $\max_{it}$ iterações, λ novos indivíduos são gerados e avaliados.

Total. $N_{fit} = \mu + \max_{it} \cdot \lambda$

3.4.6 Evolução Diferencial

Em 1995, J. Storn e K. Price propuseram um novo algoritmo dedicado à otimização de funções de parâmetros reais denominado Evolução Diferencial (em inglês, *Differential Evolution* (DE)) (Storn e Price, 1995) (Storn e Price, 1997). Uma característica peculiar desta técnica é o uso de informação presente na própria população para introduzir perturbações nos indivíduos, em contraste com o método tradicional empregado em outros algoritmos evolutivos, nos quais as perturbações estão associadas a distribuições de probabilidade pré-determinadas.

Em suma, DE utiliza N_P vetores de parâmetros M-dimensionais $\boldsymbol{\varphi}_{i,G}, i = 1, \ldots, N_P$, como população em cada geração G. Novos vetores de parâmetros são gerados através da adição da diferença ponderada entre dois vetores de parâmetros a um terceiro indivíduo. Considere esta operação como uma mutação (Storn e Price, 1997) (Price, Storn, e Lampinen, 2005).

Os vetores de parâmetros mutados são então combinados com outros vetores pré-determinados, denominados *target vectors*, a fim de gerar os *trial vectors*. Esta combinação de parâmetros é referida como *crossover* em DE. Caso o *trial vector* forneça um valor de *fitness* maior (maximização) que aquele associado ao respectivo *target vector*, este último dará lugar ao primeiro na próxima geração. Esta operação corresponde à seleção.

Conforme já mencionado na Seção 3.4.1, este procedimento se repete até que um número máximo de iterações $\max_{it}$ seja atingido. Note que, embora tenha surgido a partir de uma

motivação bastante diferente e manipule os indivíduos da população de uma forma distinta, DE também se caracteriza pelo uso da estrutura básica de um algoritmo evolutivo, como a apresentada no início da Seção 3.4.

A seguir, descrevemos com mais detalhes os operadores de mutação, *crossover* e seleção empregados pelo algoritmo de evolução diferencial.

Operador de Mutação

Para cada indivíduo da população (*target vector*) $\boldsymbol{\varphi}_{i,G}$, $i = 1, \ldots, N_P$, um novo vetor é gerado por meio da seguinte relação:

$$\mathbf{v}_{i,G+1} = \boldsymbol{\varphi}_{r_1,G} + F \cdot (\boldsymbol{\varphi}_{r_3,G} - \boldsymbol{\varphi}_{r_2,G}), \tag{3.20}$$

onde r_1,r_2,$r_3 \in \{1, 2, \ldots, N_P\}$ são índices aleatoriamente escolhidos, mutuamente distintos e também diferentes do índice i. A constante real $F \in [0, 2]$ determina o tamanho do passo a ser dado na direção definida pelo vetor diferença $\boldsymbol{\varphi}_{r_3,G} - \boldsymbol{\varphi}_{r_2,G}$.

A Figura 3.1 ilustra como o vetor $\mathbf{v}_{i,G+1}$ é gerado por meio do processo de mutação no caso bidimensional.

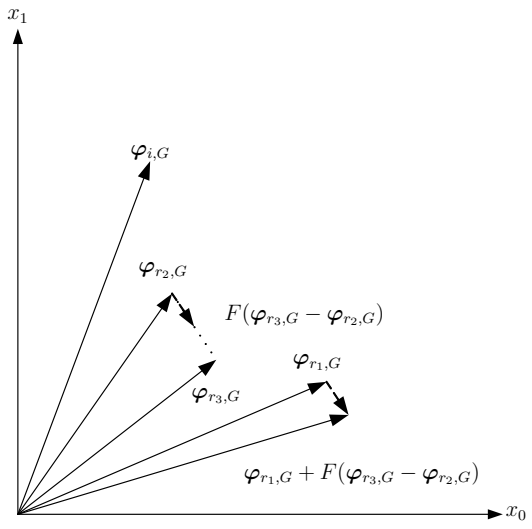


Figura 3.1: Exemplo bidimensional do processo de mutação.

Operador de Crossover

Com a finalidade de aumentar a diversidade dos vetores de parâmetros mutados, um procedimento similar ao *crossover* é utilizado. Seja $\boldsymbol{\varphi}_{i,G}$ o *target vector* sob análise e $\mathbf{v}_{i,G+1}$ o respectivo vetor mutado obtido por meio da Expressão (3.20). O vetor $\mathbf{u}_{i,G+1} = (u_{1i,G+1}\, u_{2i,G+1} \ldots u_{Mi,G+1})$, denominado *trial vector*, é obtido da seguinte maneira:

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1}, & \text{se } r_j \leq CR \text{ ou } j = I_i \\ \varphi_{ji,G}, & \text{se } r_j > CR \text{ e } j \neq I_i \end{cases}, \tag{3.21}$$

onde $j = 1, \ldots, M$, $r_j \backsim \mathcal{U}(0,1)$, $CR \in [0,1]$ é uma constante que define a taxa de *crossover* e I_i é um índice aleatoriamente escolhido $\in \{1, \ldots, M\}$, o que garante que $\mathbf{u}_{i,G+1}$ recebe pelo menos uma componente de $\mathbf{v}_{i,G+1}$. A Figura 3.2 esboça o processo de geração do *trial vector* $\mathbf{u}_{i,G+1}$ via *crossover*.

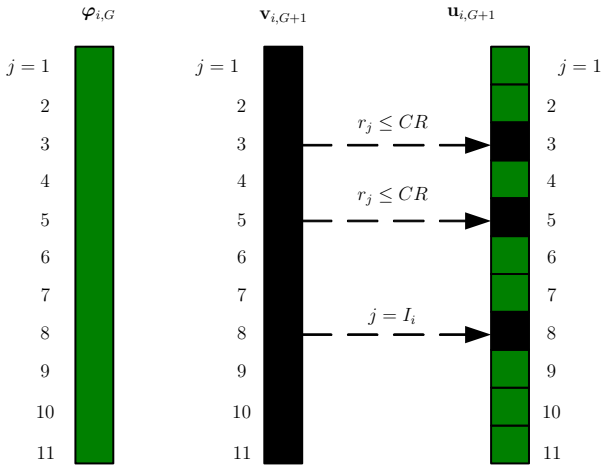


Figura 3.2: Exemplo do processo de *crossover*.

Operador de Seleção

Após as etapas de mutação e *crossover*, nas quais todos os N_P vetores serviram como *target vector*, a seleção dos vetores que serão preservados para a próxima geração é feita usando o

seguinte critério: Seja $\boldsymbol{\varphi}_{i,G}$ o *target vector* sob análise e $\mathbf{u}_{i,G+1}$ seu respectivo *trial vector*,

- Se o *fitness* de $\mathbf{u}_{i,G+1}$ for superior ao *fitness* de $\boldsymbol{\varphi}_{i,G}$, então $\boldsymbol{\varphi}_{i,G+1} = \mathbf{u}_{i,G+1}$.
- Caso contrário, $\boldsymbol{\varphi}_{i,G+1} = \boldsymbol{\varphi}_{i,G}$.

A fim de facilitar a discriminação das principais variantes de DE, a notação DE/x/y/z foi introduzida, onde:

1. x - especifica o vetor a ser mutado, isto é, $\mathbf{x}_{r_1,G}$.
2. y - determina o número de vetores diferença (direções) utilizados na etapa de mutação.
3. z - indica o esquema de *crossover* adotado.

A abreviação DE/rand/1/bin corresponde ao algoritmo apresentado nas seções anteriores e representa a proposta clássica da evolução diferencial (Storn e Price, 1997). O Quadro 6 exibe o pseudo-código do algoritmo de evolução diferencial empregado neste trabalho.

Quadro 6 Evolução Diferencial

```
Função φ = DE(N_P,M,CR,F,range,F_fit)
[φ, f_φ] ⇐ inicializa(N_P,M,range,F_fit)
Enquanto critério de parada não for satisfeito faça
   Para i = 1 até N_P faça
      v_{i,G+1} ⇐ mutação(φ_{i,G},F)
      u_{i,G+1} ⇐ crossover(φ_{i,G},v_{i,G+1},CR)
   Fim Para
   f_u ⇐ avalie(u,F_fit)
   Para i = 1 até N_P faça
      Se f_u(i) > f_φ(i) Então
         φ_{i,G+1} ⇐ u_{i,G+1}
      Senão
         φ_{i,G+1} ⇐ φ_{i,G}
      Fim Se
   Fim Para
Fim Enquanto
```

Número de avaliações da função de fitness

A partir da descrição do algoritmo de evolução diferencial feita nas seções anteriores, podemos deduzir uma expressão para o número de vezes que a função de *fitness* é avaliada:

1. População inicial: são realizadas N_P avaliações.

2. Em cada uma das $\max_{it}$ iterações, N_P novos indivíduos são gerados e avaliados.

Total. $N_{fit} = N_P + \max_{it} \cdot N_P$

3.5 Sistemas Imunológicos Artificiais

O sistema imunológico dos animais vertebrados é uma coleção de células, moléculas e organismos distribuídos que cooperam dinamicamente para a manutenção de um estado de equilíbrio interno em seus corpos (de Castro, 2006). Este complexo sistema possui habilidades formidáveis, dentre as quais destacam-se sua capacidade de reconhecer sinais internos e externos e de aprender a enfrentar doenças causadas por agentes invasores.

Sistemas imunológicos artificiais (em inglês, *artificial immune systems* (AIS)) é a terminologia introduzida que se refere a qualquer sistema adaptativo que, tendo por base a teoria imunológica, tem por objetivo a solução de problemas (Castro e Timmis, 2002).

Um dos principais elementos teóricos desenvolvidos no estudo dos sistemas imunológicos é o princípio de seleção clonal (Burnet, 1959), o qual é utilizado para explicar os aspectos fundamentais de uma resposta imunológica adaptável a um estímulo antigênico. Juntamente com a teoria da maturação de afinidade (Ada e Nossal, 1987), a seleção clonal forma o núcleo de uma resposta imunológica adaptativa. Ambas teorias têm sido empregadas na literatura de sistemas imunológicos artificiais para o projeto de sistemas adaptativos com a finalidade de resolver problemas em diferentes áreas, como reconhecimento de padrões e otimização.

Em síntese, o princípio de seleção clonal estabelece a idéia de que somente as células capazes de reconhecer um determinado antígeno, isto é, as células de maior afinidade, proliferam, sendo, portanto, selecionadas em detrimento daquelas que não reconhecem o antígeno (Burnet, 1959). Estas células são estimuladas a se reproduzir, gerando cópias (clones) de si mesmas, de maneira proporcional à afinidade: quanto maior a afinidade, maior o número de clones gerados.

Como em todos os eventos reprodutivos, este processo de clonagem também está sujeito a erros (mutações). Contudo, uma característica peculiar do sistema imunológico é que as

mutações ocorrem numa taxa inversamente proporcional à afinidade entre estas células e o antígeno: se a afinidade entre o antígeno e o anticorpo for alta, então a taxa de mutação é baixa; em contrapartida, quando a afinidade é baixa, a taxa de mutação é alta. Este processo de adaptação das células baseado na afinidade é chamado de maturação de afinidade (Ada e Nossal, 1987). Por fim, um mecanismo de seleção permite que as células mutadas que apresentam maior afinidade com o antígeno sejam guardadas em um repertório de memória.

Intuitivamente, podemos compreender como estes processos contribuem para a adaptação da resposta do sistema imunológico a um determinado antígeno: as células de maior afinidade geram um número maior de clones, os quais sofrem mutações segundo uma taxa pequena, de modo que tendem a ser bem semelhantes às células originais. Logo, devem também possuir uma afinidade alta com aquele antígeno.

Por outro lado, as células com pouca afinidade geram um número menor de clones. Estes, por sua vez, sofrem mutações com elevadas taxas, de modo que tendem a ser bastante diferentes das células originais, o que pode ser interessante, já que estas últimas não apresentavam boa capacidade de reconhecer e enfrentar o antígeno. De todo o repertório de clones gerados, somente aqueles de maior afinidade é que são selecionados e armazenados em um conjunto de memória, o que certamente favorece o aprimoramento do sistema imunológico no combate a este antígeno.

Observe que o funcionamento do sistema imunológico, segundo os princípios de seleção clonal e maturação de afinidade, pode ser interpretado como uma instância do processo evolutivo, uma vez que exibe os três principais requisitos para evolução: diversidade, variação genética e seleção natural (Cziko, 1995).

Estas teorias imunológicas serviram de inspiração para o desenvolvimento de diversas ferramentas computacionais, as quais procuram extrair e reproduzir algumas características inerentes aos sistemas imunológicos naturais para a solução de problemas. Apesar de sua diversidade, a maioria dos AIS compartilha algumas propriedades interessantes:

- São inerentemente capazes de manter diversidade na população.
- Tendem a preservar soluções localmente ótimas quando as encontram.

Estas características tornam os AIS excelentes candidatos para lidar com problemas de otimização, e, de fato, alguns algoritmos imuno-inspirados já foram desenvolvidos para este propósito (de Castro e Von Zuben, 2002) (de Castro, 2006). A seguir, descrevemos os dois algoritmos imuno-inspirados que foram aplicados ao problema de estimação DOA.

3.5.1 CLONALG

Proposto por L. N. de Castro e F. J. Von Zuben, em 2002, o algoritmo CLONALG utiliza os princípios de seleção clonal e maturação de afinidade como os principais mecanismos para manipular uma população de soluções candidatas, denominadas anticorpos (de Castro e Von Zuben, 2002). Inicialmente destinado a tarefas de reconhecimento de padrões, este algoritmo também foi adaptado para lidar com problemas de otimização multimodal. A seguir, apresentamos uma descrição básica do algoritmo CLONALG adaptado para esta última tarefa (de Castro, 2006):

1. Inicialização: crie uma população inicial **Ab** contendo N_P anticorpos.
2. Avaliação: calcule o *fitness* de cada elemento de **Ab**.
3. Seleção Clonal e Expansão: selecione os n_1 anticorpos de maior *fitness* em **Ab** e gere clones destes anticorpos de maneira proporcional ao *fitness* - quanto maior o *fitness*, maior o número de clones, e vice-versa.
4. Maturação de Afinidade: aplique mutação sobre os clones segundo uma taxa inversamente proporcional ao *fitness* - quanto maior o *fitness*, menor a taxa de mutação, e vice-versa. Então, avalie a população de clones e forme a nova população **Ab** com os n_1 melhores clones [4].

[4] É comum manter um clone sem sofrer mutação a fim de evitar a perda do melhor anticorpo.

5. Metadinâmica: substitua uma porcentagem p_{Ab} dos anticorpos de menor *fitness* em $\mathbf{Ab}$ por novos gerados aleatoriamente. Este procedimento pode ser feito em períodos de T_{Ab} iterações.

6. Ciclo: repita os passos 2 a 5 até que um determinado critério de parada seja atingido.

Quando o processo de otimização busca encontrar múltiplos ótimos a partir de uma única população de anticorpos, algumas recomendações quanto à escolha de alguns parâmetros são dadas: 1) $n_1 = N_P$, ou seja, todos os anticorpos em $\mathbf{Ab}$ são selecionados para a etapa de clonagem; 2) o número de clones (N_c) gerados para cada anticorpo é o mesmo, ou seja, não mais depende do *fitness*. Em geral, define-se $N_c = \psi N_P$, onde $\psi \in [0, 1]$.

Embora a versão inicial do algoritmo CLONALG trabalhe com uma representação binária para os anticorpos, neste trabalho foram feitas algumas adaptações para o caso em que os anticorpos correspondem a vetores reais, uma vez que esta é a forma mais simples e direta de representar as estimativas DOA. A principal modificação ocorre na etapa de mutação: Seja $\mathbf{C}^{(\mathbf{i})}_{\mathbf{Ab}}$ um clone do i-ésimo anticorpo presente na população $\mathbf{Ab}$. A operação de mutação de afinidade consiste na adição de uma variável aleatória com distribuição normal de média zero e desvio padrão regulado por um parâmetro ρ e por um fator inversamente proporcional ao *fitness* de $\mathbf{Ab_i}$, conforme as expressões

$$\bar{\mathbf{Ab}}_{\mathbf{i}} = \mathbf{C}^{(\mathbf{i})}_{\mathbf{Ab}} + \alpha_m N(0, 1) \tag{3.22}$$

$$\alpha_m = \frac{1}{\rho} e^{-fAb_i^*}, \tag{3.23}$$

onde $\bar{\mathbf{Ab}}_{\mathbf{i}}$ representa o anticorpo mutado e fAb_i^* denota o *fitness* do i-ésimo anticorpo normalizado no intervalo $(0, 1)$.

O Quadro 7 exibe o pseudo-código do algoritmo CLONALG utilizado neste trabalho para estimar os ângulos de chegada.

Quadro 7 Algoritmo CLONALG

```
Função [Ab] = CLONALG(N_P,M,N_c,range,ρ,T_Ab,p_Ab,max_it,F_fit)
[Ab, fAb] ⇐ inicializa(N_P,M,range,F_fit)
Para it = 1 até max_it faça
   Para i = 1 até N_P faça
      C^(i)_Ab ⇐ clone(Ab_i,N_c)
      Ab̄_i ⇐ mutação(C^(i)_Ab,fAb_i,N_c,ρ)
      fC ⇐ avalie(Ab̄_i,F_fit)
      [f_max, imax] ⇐ max (fC)
      Se f_max > fAb_i Então
         Ab_i ⇐ Ab_i(imax)
         fAb_i ⇐ f_max
      Fim Se
   Fim Para
   Se it é múltiplo de T_Ab Então
      [Ab, fAb] ⇐ insere(Ab,fAb,p_Ab,N_P,M,range,F_fit)
   Fim Se
Fim Para
```

Número de avaliações da função de fitness

Inicialmente, são realizadas N_P avaliações por causa da população inicial. Em cada iteração, são gerados e avaliados $N_c \cdot N_P$ clones. Além disto, em períodos de T_{Ab} iterações, $\lfloor p_{Ab} \cdot N_P \rfloor$ novos anticorpos são introduzidos na população e também são avaliados. Logo, o número total de avaliações da função de *fitness* para o algoritmo CLONALG pode ser expresso por:

$$N_{fit} = N_P + \max_{it} N_P N_c + \left\lfloor \frac{\max_{it}}{T_{Ab}} \right\rfloor \cdot \lfloor p_{Ab} N_P \rfloor . \tag{3.24}$$

3.5.2 Opt-aiNet

Ainda em 2002, L. N. de Castro e J. Timmis propuseram uma rede imunológica artificial destinada à otimização de funções de variáveis reais - a opt-aiNet (de Castro e Timmis, 2002). Essa rede, que une as idéias de seleção clonal e maturação de afinidade (de Castro e Von Zuben, 2002) à noção de rede imunológica (Jerne, 1974), caracteriza-se por um elegante compromisso entre exploração e explotação do espaço de busca, o que abre perspectivas interessantes em domínios multimodais.

A teoria da rede imunológica, formalizada por N. Jerne, sugere que todas as células e moléculas presentes no sistema imunológico são capazes de reconhecerem-se mutuamente. Ao

contrário da visão proposta pela teoria da seleção clonal, segundo a qual o sistema imunológico seria composto por células e moléculas em descanso, sendo, portanto, estimulado unicamente por antígenos externos, a teoria da rede traz a visão de um sistema imunológico dinâmico, cujo comportamento é regido pelos padrões de interação entre as células imunológicas, e cujo estado se altera devido a perturbações originadas por estímulos antigênicos (Jerne, 1974).

A rede opt-aiNet emprega os mesmos processos de seleção clonal e maturação por afinidade que o algoritmo CLONALG. Além disto, a opt-aiNet incorpora a noção de rede imunológica por meio da introdução de um controle dinâmico do tamanho da população, o qual é responsável por mecanismos de supressão de anticorpos similares e pela introdução de novos anticorpos (de Castro e Timmis, 2002). Com isto, a diversidade da população é mantida e consegue-se explorar adequadamente o espaço de busca.

De maneira similar às técnicas de *niching* discutidas nas Seções 3.4.2 a 3.4.4, o conceito de similaridade entre indivíduos está associado a uma métrica de distância: considera-se que dois indivíduos são similares quando a distância entre eles é inferior a um limiar de similaridade σ_s.

Outra particularidade da rede opt-aiNet é que as operações de supressão e inserção de anticorpos são realizadas somente quando se identifica a ocorrência de estagnação na população: considera-se que a população está estagnada quando a variação percentual do *fitness* médio da população entre duas iterações é inferior a valor pequeno F_{dif}. Convém destacar que esta verificação ocorre apenas em períodos de T_{Ab} iterações. Desta forma, permite-se que a população evolua por um período, através de sucessivas operações de seleção clonal e mutação de afinidade, antes de se executar a operação de supressão.

O Quadro 8 apresenta o pseudo-código da rede opt-aiNet empregada neste trabalho para a estimação dos ângulos de chegada.

Podemos observar no Quadro 8 que a principal distinção entre a rede opt-aiNet e o algoritmo CLONALG reside na introdução dos mecanismos de supressão e inserção quando se verifica a ocorrência de estagnação na população. As primeiras etapas executadas pela opt-aiNet estão associadas aos processos de seleção clonal e maturação de afinidade, como também ocorre com

Quadro 8 Rede opt-aiNet

Função [**Ab**] = opt-aiNet(N_P,M,N_c,range,ρ,σ_s,T_{Ab},p_{Ab},F_{dif},max$_{it}$,F_{fit})
[**Ab**, **fAb**] ⇐ inicializa(N_P,M,range,F_{fit})
Para $it = 1$ até max$_{it}$ **faça**
  Para $i = 1$ até N_P **faça**
    $\mathbf{C}^{(\mathbf{i})}_{\mathbf{Ab}}$ ⇐ clone($\mathbf{Ab_i}$,N_c)
    $\bar{\mathbf{Ab}}_{\mathbf{i}}$ ⇐ mutação($\mathbf{C}^{(\mathbf{i})}_{\mathbf{Ab}}$,$fAb_i$,$N_c$,$\rho$)
    fC ⇐ avalie($\bar{\mathbf{Ab}}_{\mathbf{i}}$,$F_{fit}$)
    $[f_{max}, imax]$ ⇐ max (**fC**)
    Se $f_{max} > fAb_i$ **Então**
      $\mathbf{Ab_i}$ ⇐ $\mathbf{Ab_i}(\bar{\mathbf{i}}\mathbf{max})$
      $fAb_i \Leftarrow f_{max}$
    Fim Se
  Fim Para
  Se it é múltiplo de T_{Ab} **Então**
    Se variação percentual do *fitness* médio é inferior a F_{dif} **Então**
      N_i ⇐ número de indivíduos em **Ab**
      [**Ab**, **fAb**] ⇐ supressão(**Ab**,**fAb**,σ_s)
      [**Ab**, **fAb**] ⇐ insere(**Ab**,**fAb**,p_{Ab},N_i,M,range,F_{fit})
      N_P ⇐ número de indivíduos em **Ab**
    Fim Se
  Fim Se
Fim Para

o algoritmo CLONALG, enquanto a última parte está relacionada à idéia de rede imunológica.

Número de avaliações da função de fitness

Uma vez que a rede opt-aiNet ajusta dinamicamente o tamanho da população de anticorpos, e que este ajuste é feito com base no grau de similaridade entre anticorpos, dado pela distância entre eles no espaço de busca, não é possível derivar uma expressão para o número de vezes que a função de *fitness* é avaliada. Contudo, para uma dada iteração i, pode-se apontar o número de avaliações de *fitness* em cada etapa do algoritmo, conforme mostramos a seguir:

1. Na etapa de mutação, são realizadas $N_c \cdot N_P(i)$ avaliações, onde $N_P(i)$ indica o número de anticorpos presentes na população na iteração i.

2. Caso haja supressão de indivíduos, em seguida é feita a inserção e a avaliação de $\lfloor p_{Ab} N_P(i) \rfloor$ novos anticorpos.

3.6 Inteligência de Enxame

Na natureza, encontramos diversos exemplos de espécies que se beneficiam do convívio em sociedade de modo a obter vantagens para sua sobrevivência. A vida em sociedade pode propiciar um aumento nas chances de reprodução, reduzir a probabilidade de ataque de predadores, facilitar a localização e coleta de alimento e também permitir a divisão de tarefas entre os indivíduos da sociedade. Estes comportamentos sociais também serviram de inspiração para o desenvolvimento de ferramentas computacionais para a solução de problemas.

O termo inteligência de enxame (em inglês, *swarm intelligence*) refere-se a uma propriedade de sistemas compostos por agentes não-inteligentes com capacidades individuais limitadas que interagem entre si e com o meio no qual estão inseridos e que exibem comportamentos coletivos inteligentes (White e Pagurek, 1998) (de Castro, 2006).

De maneira sucinta, os sistemas baseados em inteligência de enxame fazem uso de uma população (enxame) de indivíduos (agentes) capazes de interagir, direta ou indiretamente, com o ambiente e entre si. Como resultado destas interações, podem ocorrer mudanças no ambiente ou nos próprios indivíduos, o que pode levar ao surgimento de comportamentos emergentes úteis para uma determinada finalidade. A seguir, descrevemos os algoritmos de inteligência de enxame que foram aplicados ao problema de estimação DOA.

3.6.1 Particle Swarm

Em 1995, Kennedy e Eberhart propuseram o algoritmo denominado enxame de partículas (em inglês, *particle swarm* (PS)) a partir da idéia de reproduzir em computador a forma pela qual as sociedades humanas processam o conhecimento (Kennedy e Eberhart, 1995) (Kennedy, 1997).

O *particle swarm* baseia-se nos princípios de uma teoria sociocognitiva muito simples (Kennedy, Eberhart, e Shi, 2001) (Kennedy, 2004). Os autores afirmam que o processo de adaptação social compreende uma componente de baixo nível, que corresponde ao comporta-

mento real dos indivíduos (partículas), e uma componente de alto nível, que está associada à formação de padrões entre os indivíduos. Pode-se dizer, portanto, que: 1) cada indivíduo de uma população possui sua própria experiência e é capaz de avaliar sua qualidade; 2) como há interação social entre os indivíduos, eles também possuem conhecimentos sobre os desempenhos de seus vizinhos. Estes dois tipos de informação correspondem ao aprendizado individual e à transmissão social ou cultural, respectivamente.

Três princípios também podem ser utilizados para descrever de maneira resumida o processo de adaptação social (Kennedy et al., 2001), os quais podem ser combinados de modo a possibilitar a adaptação dos indivíduos ao ambiente:

1. Avaliação: a habilidade dos indivíduos de sensorear o ambiente permite que estes avaliem sua qualidade em relação a algum parâmetro ou objetivo. Por exemplo, a altura e o peso podem ser utilizados como parâmetros para avaliar a capacidade de uma pessoa se tornar um ginasta profissional.

2. Comparação: as pessoas usualmente tomam outros indivíduos como padrões para avaliarem a si próprios, o que pode servir como uma forma de motivação para aprender e mudar. Por exemplo, podemos olhar para outras pessoas a fim de avaliar nossa aparência e personalidade.

3. Imitação: a imitação realizada pelo ser humano corresponde a incorporar a perspectiva de outra pessoa, não apenas pela reprodução de um determinado comportamento, mas pela compreensão de seu propósito e a execução deste comportamento quando apropriado.

Estes princípios também estão presentes no algoritmo PS, pois os indivíduos da população, durante a busca pela solução de um determinado problema, aprendem a partir de suas experiências no passado e das experiências de outros indivíduos. Cada indivíduo, além de avaliar a si próprio, se compara com seus vizinhos e imita aqueles cujo desempenho for superior ao seu.

Em termos matemáticos, a posição de cada indivíduo (partícula) é dada por $\mathbf{x_i}$, que corresponde a um vetor no espaço dos atributos $\mathcal{R}^M$. A partícula se moverá com uma "velocidade vetorial" $\boldsymbol{\nu_i}$ de modo que sua próxima posição é dada por:

$$\mathbf{x_i}(t+1) = \mathbf{x_i}(t) + \boldsymbol{\nu_i}(t+1). \tag{3.25}$$

O vetor velocidade $\boldsymbol{\nu_i}$, que indica a direção do movimento da partícula, é função da posição atual da partícula ($\mathbf{x_i}(t)$), do vetor velocidade na iteração anterior, da melhor posição já visitada pela partícula até o momento ($\mathbf{p_i}$) e pela melhor posição encontrada por qualquer membro de sua vizinhança ($\mathbf{p_g}$), como mostra a expressão:

$$\boldsymbol{\nu_i}(t+1) = \boldsymbol{\nu_i}(t) + \boldsymbol{\psi_1} \otimes (\mathbf{p_i}(t) - \mathbf{x_i}(t)) + \boldsymbol{\psi_2} \otimes (\mathbf{p_g}(t) - \mathbf{x_i}(t)), \tag{3.26}$$

onde $\boldsymbol{\psi_1}$ e $\boldsymbol{\psi_2}$ são vetores aleatórios positivos cujos elementos são extraídos de uma distribuição uniforme com limitante superior pré-definido: $\boldsymbol{\psi_1} \backsim \mathcal{U}(0, L_1)$ e $\boldsymbol{\psi_2} \backsim \mathcal{U}(0, L_2)$, sendo que L_1 e L_2 são denominados constantes de aceleração [5]. O operador $\otimes$ denota uma multiplicação elemento a elemento entre vetores.

Observe na Expressão (3.26) que a parcela $(\mathbf{p_g}(t) - \mathbf{x_i}(t))$ explicita a interação social entre a partícula e seus vizinhos, ao passo que o termo $(\mathbf{p_i}(t) - \mathbf{x_i}(t))$ representa a parcela cognitiva.

A fim de limitar as variações na posição da partícula, de modo a evitar que o sistema se expanda indefinidamente, dois valores ν_{min} e ν_{max} são definidos para os elementos do vetor velocidade $\boldsymbol{\nu}$, garantindo assim que a partícula oscile dentro de limites pré-estabelecidos.

O conceito de vizinhança não necessariamente significa similaridade no espaço dos parâmetros. Na verdade, refere-se a uma similaridade topológica associada a uma determinada estrutura ou arranjo. Desta forma, cada partícula terá acesso ao desempenho e posição de seus vizinhos estabelecidos pela topologia adotada. Existem diversas formas de se definir a

[5] Segundo Kennedy (Kennedy, 2004), a soma das constantes de aceleração deve ser igual a $L_1 + L_2 = 4{,}1$, sendo o procedimento mais comum adotar $L_1 = L_2 = 2{,}05$.

vizinhança, dentre as quais destacamos a vizinhança em anel (cada partícula tem um vizinho à direita e à esquerda, fechando um anel) e a vizinhança totalmente conectada (todas as partículas são vizinhas entre si) (de Castro, 2006).

O Quadro 9 apresenta o pseudo-código do algoritmo *particle swarm* empregado neste trabalho para a estimação dos ângulos de chegada.

Quadro 9 Algoritmo *Particle Swarm*

Função [**Ab**] = PS(N_P,M,range,ν_{min},ν_{max},L_1,L_2,F_{fit})
[**Pop**, **fPop**, $\boldsymbol{\nu}$] $\Leftarrow$ inicializa(N_P,M,range,ν_{min},ν_{max},F_{fit})
Enquanto critério de parada não for satisfeito **faça**
  Para $i = 1$ até N_P **faça**
    atualiza a melhor posição da partícula ($\mathbf{p_i}$) e a melhor posição entre os vizinhos ($\mathbf{p_g}$)
    $\boldsymbol{\nu_i} \Leftarrow \boldsymbol{\nu_i} + \boldsymbol{\psi_1} \otimes (\mathbf{p_i} - \mathbf{Pop_i}) + \boldsymbol{\psi_2} \otimes (\mathbf{p_g} - \mathbf{Pop_i})$
    $\boldsymbol{\nu_i} \in [\nu_{min}, \nu_{max}]$
    $\mathbf{Pop_i} \Leftarrow \mathbf{Pop_i} + \boldsymbol{\nu_i}$
  Fim Para
Fim Enquanto

3.6.2 Particle Swarm com fator de inércia

Em 1999, Shi e Eberhart introduziram uma modificação ao algoritmo *particle swarm*. Eles sugeriram o uso de um fator denominado peso de inércia (ζ) com o propósito de balancear a exploração e a explotação do espaço de busca (Shi e Eberhart, 1999). Os autores apontam os papéis exercidos por cada fator da Expressão (3.26) na atualização do vetor velocidade: 1) o primeiro termo favorece a expansão do espaço de busca que uma partícula pode explorar, isto é, contribui para a exploração de novas regiões. 2) os dois termos seguintes contribuem para a exploração local de certas regiões, permitindo a contração do espaço de busca.

Estas duas partes juntas conferem ao algoritmo a possibilidade de explorar o espaço de busca tanto globalmente quanto localmente. Contudo, deve ser feito um balanceamento entre busca local e global para que o algoritmo consiga efetivamente realizar a tarefa de encontrar o ótimo global. Além disto, este balanceamento deve ser diferente quando consideramos problemas distintos, uma vez que as características dos respectivos espaços de busca não são necessariamente semelhantes.

Tendo isto em mente, o peso de inércia foi introduzido, de forma que a expressão de atualização do vetor velocidade é dada por:

$$\boldsymbol{\nu_i}(t+1) = \zeta\boldsymbol{\nu_i}(t) + \boldsymbol{\psi_1} \otimes (\mathbf{p_i}(t) - \mathbf{x_i}(t)) + \boldsymbol{\psi_2} \otimes (\mathbf{p_g}(t) - \mathbf{x_i}(t)). \tag{3.27}$$

3.6.3 Particle Swarm com termo de constrição

Outra variante do algoritmo *particle swarm* original foi proposta por Clerc e Kennedy em 2002 (Clerc e Kennedy, 2002). Agora, o vetor velocidade de uma partícula é determinado segundo a expressão:

$$\boldsymbol{\nu_i}(t+1) = \chi\left\{\boldsymbol{\nu_i}(t) + \boldsymbol{\psi_1} \otimes (\mathbf{p_i}(t) - \mathbf{x_i}(t)) + \boldsymbol{\psi_2} \otimes (\mathbf{p_g}(t) - \mathbf{x_i}(t))\right\}, \tag{3.28}$$

onde χ é conhecido como coeficiente de constrição[6]. Com a introdução do termo χ, não há necessidade de se definir limites para o vetor velocidade. Além disto, foi verificado que, a depender dos valores das constantes de aceleração L_1 e L_2, a escolha de χ pode atenuar a ocorrência de comportamentos indesejáveis, como explosão, e até mesmo contribuir para a convergência do enxame de partículas.

Número de avaliações da função de fitness

Os três algoritmos baseados em inteligência de enxame, descritos nas seções anteriores, diferem apenas na expressão utilizada para o cálculo do vetor velocidade de cada partícula, de modo que o número de avaliações da função de *fitness* é equivalente. Apresentamos abaixo a expressão derivada para o número médio de avaliações da função de *fitness* associada a estes algoritmos:

1. População inicial: são realizadas N_P avaliações.

[6]O valor $\chi \approx 0{,}729$ é sugerido pelos autores no caso em que $L_1 = L_2 = 2{,}05$

2. Em cada uma das $\max_{it}$ iterações, as posições das N_P partículas são atualizadas. Por isso, a população inteira é re-avaliada.

Total. $N_{fit} = N_P + \max_{it} \cdot N_P$

3.7 Epílogo

Neste capítulo, apresentamos os algoritmos de computação natural a serem empregados no problema de estimação de direção de chegada como alternativa aos métodos clássicos descritos no Capítulo 2. Inicialmente, diante das características desafiadoras inerentes à tarefa de encontrar as estimativas ML para os ângulos de chegada e, à luz do teorema *No-Free-Lunch*, foi possível apontar as razões que sustentam e motivam a aplicação desta classe de algoritmos a este problema.

A partir dos principais ramos de pesquisa dentro de computação natural, a saber, computação evolutiva, sistemas imunológicos artificiais e inteligência de enxame, foi selecionado um conjunto de algoritmos levando em consideração os critérios discutidos na Seção 3.2. Cada ramo de pesquisa foi brevemente apresentado e, concomitantemente, cada algoritmo foi descrito detalhadamente. Os principais parâmetros associados a cada ferramenta foram expostos, bem como os operadores utilizados. Quando pertinente, exibimos o pseudo-código do algoritmo.

O próximo passo, portanto, é avaliar o desempenho de cada algoritmo no problema de estimação DOA. No próximo capítulo, apresentaremos em detalhes o processo de ajuste dos algoritmos de computação natural ao problema DOA, além de analisarmos o desempenho destas ferramentas considerando um cenário clássico na literatura DOA.

Capítulo 4

Análise: Estudo de Caso

4.1 Introdução

O primeiro passo para a aplicação de algoritmos de computação natural a um problema em particular consiste na especificação da representação e na definição de uma função de avaliação (*fitness*). Estes elementos estabelecem a ponte entre o problema original e o procedimento adotado para sua solução. Na Seção 3.3, tais conceitos foram definidos para o problema de estimação DOA.

O passo seguinte corresponde à escolha dos principais componentes de cada algoritmo, tais como operadores de variação (mutação e *crossover*, por exemplo) adaptados à representação adotada e mecanismos de seleção de indivíduos. No capítulo 3, os algoritmos de computação natural e todos os componentes utilizados neste trabalho foram apresentados detalhadamente.

Pode-se perceber que cada algoritmo possui um conjunto de parâmetros específico que necessita ser ajustado. Por exemplo, o algoritmo genético descrito na Seção 3.4.1 requer a definição das probabilidades de *crossover* e mutação, além do número de indivíduos participantes em cada torneio e também presentes na própria população.

Os valores atribuídos aos parâmetros são cruciais para determinar se um algoritmo obterá ou não êxito na tarefa de encontrar o mínimo global da função custo J_{ML}, determinando assim

as melhores estimativas segundo o critério ML, como mostra a Expressão (2.37). Contudo, escolher os valores corretos é uma tarefa árdua. De modo geral, não existem recomendações sobre quais valores são os mais adequados e em quais situações os mesmos devem ser adotados. Por isso, é preciso atribuí-los de maneira particular e, muitas vezes, heurística.

Conforme já apontado na Seção 2.4, o problema de estimação DOA, mediante o critério ML, requer a solução de diversos problemas de otimização com características distintas mesmo considerando um único cenário. À medida que a relação sinal-ruído varia, a superfície da função custo J_{ML} altera suas características, conforme ilustrado na Seção 2.3. Além disto, para uma mesma SNR, diferentes experimentos, isto é, diferentes realizações dos vetores de sinal e ruído, também podem gerar superfícies bem distintas.

Estas peculiaridades do problema de estimação DOA tornam ainda mais difícil a tarefa de escolher valores para os parâmetros dos algoritmos, pois, a partir de uma única configuração, deseja-se que estes sejam capazes de lidar com os diferentes problemas de otimização, obtendo um bom desempenho em todos os casos.

Por conta disto, a fim de identificar algumas diretrizes que auxiliem o ajuste dos principais parâmetros destes algoritmos, foram realizados testes de sensibilidade envolvendo estes parâmetros, por meio dos quais tentamos visualizar isoladamente o impacto dos mesmos no desempenho de cada algoritmo no problema de estimação DOA. A seguir, descrevemos a metodologia empregada na realização dos testes de sensibilidade, bem como as principais observações e conclusões extraídas a partir deles.

4.2 Testes de Sensibilidade

4.2.1 Metodologia

É evidente que não seria viável realizar testes de sensibilidade para todos os parâmetros de cada algoritmo. Felizmente, isto também não é necessário, pois existem alguns parâmetros cujo

processo de ajuste é mais simples, já que variações em seus valores pouco afetam o desempenho geral do algoritmo, ou porque existem valores padronizados para os mesmos que se mostram suficientemente adequados em diversos contextos. Por isso, consideramos apenas os parâmetros mais críticos de cada algoritmo.

Os testes de sensibilidade foram conduzidos considerando o cenário DOA descrito na Seção 2.3: $N = 10$, $M = 2$, $\boldsymbol{\phi} = [10^\circ\ 15^\circ]^T$, $K = 100$ e $\mathbf{C} = \mathbf{I}$. Neste cenário, é possível empregarmos uma aproximação da abordagem de busca exaustiva utilizando uma grade bidimensional, conforme discutido na Seção 2.4, para obter, aproximadamente, o ótimo global da função custo J_{ML}.

Uma vez que a superfície da função custo J_{ML} varia consideravelmente à medida que a relação sinal-ruído é alterada, conforme ilustrado na Seção 2.3, os testes de sensibilidade foram realizados considerando três valores de SNR: -10, -5 e 15 dB. Os dois primeiros valores estão associados à região de limiar, conforme discutido na Seção 2.4, enquanto o último está relacionado com o comportamento assintótico.

Além disto, visto que as variações na superfície da função custo J_{ML} também podem ocorrer entre experimentos para uma mesma SNR, é preciso extrair uma média do erro de estimação, dado pela raiz quadrada do erro quadrático médio (RMSE), conforme a Expressão (2.39), a partir de um conjunto com vários experimentos a fim de se obter uma medida confiável acerca do desempenho do algoritmo. Por isso, para as SNRs de -10 e -5 dB, foram empregados 1000 experimentos, enquanto para a SNR de 15 dB, apenas 100 experimentos já são suficientes, pois, neste último caso, uma vez que a potência do ruído é muito inferior à dos sinais incidentes, a superfície da função custo sofre poucas modificações de um experimento para outro.

Sendo assim, os testes de sensibilidade referentes ao parâmetro x de um determinado algoritmo foram realizados da seguinte forma: à medida que o valor de x é modificado, monitora-se o desempenho deste algoritmo através da medida de RMSE, enquanto os demais parâmetros permanecem inalterados. Nesta etapa, utilizamos o valor RMSE fornecido pela busca em grade, considerando o mesmo conjunto de experimentos, como referência de desempenho para os al-

goritmos de computação natural.

Adicionalmente, uma vez que temos acesso a uma boa aproximação da posição do ótimo global da função custo J_{ML} em todos os experimentos, podemos medir a porcentagem de experimentos bem-sucedidos em função do parâmetro x. Considera-se que o algoritmo obteve êxito em um experimento quando a distância entre as estimativas DOA por ele obtidas e o ótimo global é suficientemente pequena. Esta forma alternativa de expressar o desempenho do algoritmo fornece uma visão complementar à métrica RMSE, a qual, por se tratar de uma média ao longo de um conjunto de experimentos, pode mascarar ou camuflar a ocorrência de falhas no processo de otimização da função custo e, conseqüentemente, na estimação dos ângulos de chegada.

Observando o comportamento das curvas de RMSE e do percentual de experimentos bem-sucedidos em função do valor do parâmetro nas três SNRs consideradas, deve-se atribuir um valor ao parâmetro x que represente uma solução de compromisso, com a qual o algoritmo é capaz de obter, aproximadamente, estimativas igualmente boas nas três condições consideradas.

Mais do que propriamente encontrar o conjunto de valores ótimos para os parâmetros, a principal finalidade dos testes de sensibilidade é verificar e analisar a influência que os mesmos exercem no comportamento de cada algoritmo.

Nas seções a seguir, apresentamos os resultados obtidos nos testes de sensibilidade feitos para os principais parâmetros dos algoritmos de computação natural.

4.2.2 Algoritmo Genético

O algoritmo genético apresentado na Seção 3.4.1 requer a definição dos seguintes parâmetros: N_P, N_t, p_c, p_m e σ_m. Além disso, é preciso estabelecer um valor para o número máximo de iterações ($\max_{it}$) do algoritmo. Tendo como base alguns experimentos preliminares, constatamos que os parâmetros N_P e N_t influenciam de modo significativo o desempenho do algoritmo e, por isso, foram então analisados por meio de testes de sensibilidade. Os demais parâmetros

foram definidos da seguinte maneira: $p_c = 0{,}5$, $p_m = 0{,}05$ e $\sigma_m = 1$. O número máximo de iterações foi $\max_{it} = 150$.

Para realizarmos os testes de sensibilidade para o parâmetro N_P, é necessário estabelecer um valor para o número de indivíduos N_t que participam em cada torneio na etapa de seleção. Semelhantemente, durante os testes para o parâmetro N_t, é preciso definir um valor para N_P. No entanto, quais devem ser estes valores? Além disso, quais devem ser as diretrizes utilizadas na escolha destes valores?

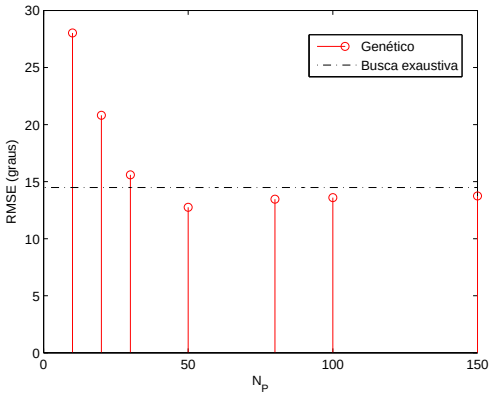
No contexto deste trabalho, esta problemática não é essencial, uma vez que o objetivo dos testes de sensibilidade, como já mencionado anteriormente, é possibilitar a identificação de alguns indicativos da influência de cada parâmetro no desempenho geral do algoritmo. Desta forma, mesmo que um valor inadequado seja atribuído a um destes parâmetros durante os testes de sensibilidade realizados para o outro, ainda assim será possível encontrar diretrizes para o ajuste deste último. Além disso, é importante enfatizar que a escolha dos valores será feita considerando os resultados dos testes de forma conjunta, de maneira a amenizar possíveis imperfeições introduzidas durante os testes.

Parâmetro N_P - Número de indivíduos

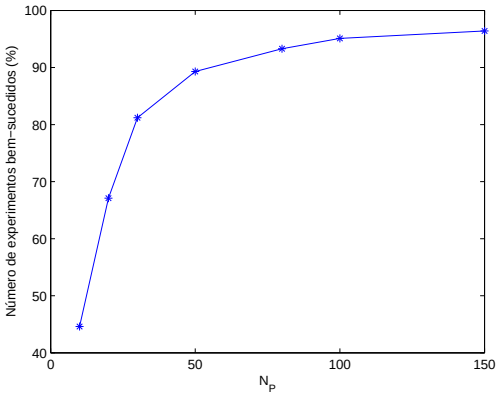
A Figura 4.1 apresenta as curvas de RMSE e do percentual de experimentos bem-sucedidos obtidas para o algoritmo genético em função do número de indivíduos da população (N_P) considerando os três valores de relação sinal-ruído mencionados na Seção 4.2.1. Em todos os experimentos, empregamos $N_t = 2$.

Em primeiro lugar, é possível observar na Figura 4.1 que, para os três valores de SNR, à medida que o número de indivíduos na população aumenta, o valor RMSE obtido pelo algoritmo genético se aproxima do valor de referência, dado pela busca em grade. Além disto, pode-se notar que a porcentagem de experimentos bem-sucedidos também aumenta conforme o valor de N_P se eleva.

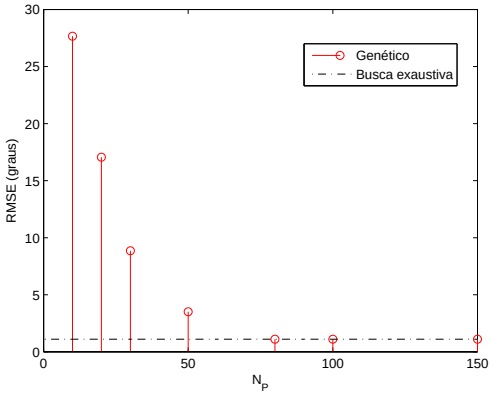
Ou seja, os resultados mostrados na Figura 4.1 indicam um progresso no desempenho do



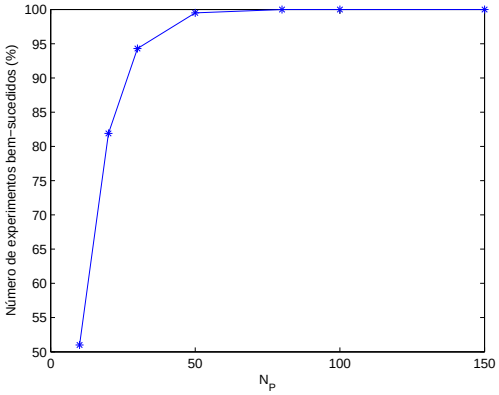
(a) SNR = −10 dB



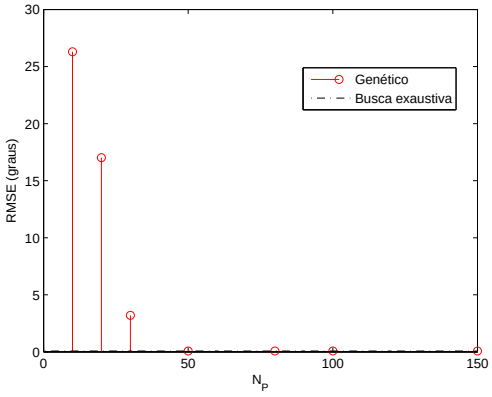
(b) SNR = −10 dB



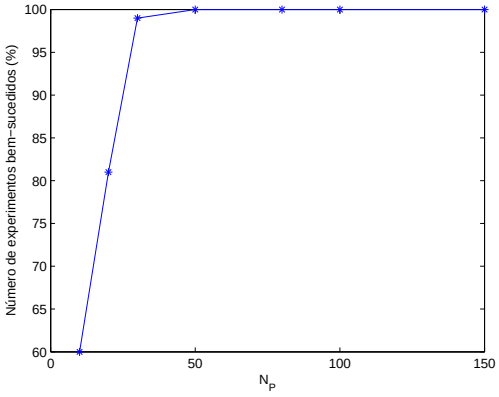
(c) SNR = −5 dB



(d) SNR = −5 dB



(e) SNR = 15 dB



(f) SNR = 15 dB

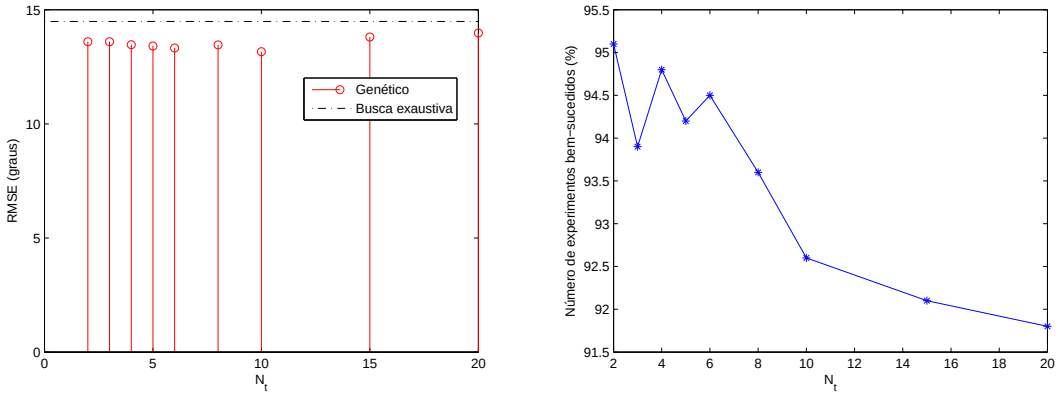
Figura 4.1: Resultados dos testes de sensibilidade para o parâmetro N_P - algoritmo genético.

algoritmo genético conforme o número de indivíduos presentes na população aumenta. Contudo, conforme deduzido na Seção 3.4.1, o número médio de avaliações da função de *fitness* referente ao algoritmo genético também cresce conforme N_P se eleva, o que representa um custo computacional maior. Logo, a escolha do valor de N_P deve ser feita visando não apenas garantir um desempenho satisfatório, mas também assegurar que o custo computacional seja suficientemente reduzido.

Um último aspecto a ser destacado encontra-se na Figura 4.1(a): para alguns valores de N_P, o algoritmo genético obteve valores de RMSE inferiores aos da busca em grade. Isto é uma forte evidência de que esta ferramenta não conseguiu localizar o ótimo global em alguns experimentos, mas acabou convergindo para pontos sub-ótimos localizados em regiões mais próximas aos valores verdadeiros dos ângulos de chegada que o próprio ótimo global, o que explica o fato de alguns valores de RMSE serem menores que os da busca em grade.

Parâmetro N_t - Número de indivíduos por torneio

A Figura 4.2 apresenta as curvas de RMSE e do percentual de experimentos bem-sucedidos obtidas para o algoritmo genético em função do número de indivíduos participantes em cada torneio (N_t) durante a etapa de seleção considerando $N_P = 100$.



(a) SNR = −10 dB (b) SNR = −10 dB

Figura 4.2: Resultados dos testes de sensibilidade para o parâmetro N_t - algoritmo genético.

Optamos por apresentar os resultados somente para a SNR de -10 dB visto que são suficientes para ilustrar a influência exercida pelo parâmetro N_t no desempenho do algoritmo. Afinal, verificamos que, no caso em que a SNR é igual a 15 dB, o impacto das variações em N_t foi praticamente desprezível e que, no caso em que a SNR é igual a -5 dB, os resultados apresentam comportamentos equivalentes aos explicitados na Figura 4.2.

A Figura 4.2(a) parece indicar que o valor de N_t pouco interfere no desempenho do algoritmo genético. Contudo, quando consideramos a Figura 4.2(b), fica claro que este parâmetro possui o seguinte efeito: à medida que aumentamos seu valor, a probabilidade de os melhores indivíduos presentes na população serem selecionados também aumenta, de maneira que a pressão seletiva sobre a população é maior, o que pode acarretar na perda de diversidade na população, inclusive de forma prematura, o que por fim contribui para que o algoritmo falhe na tarefa de localizar o ótimo global da função custo J_{ML}. Por isso, é recomendável utilizar valores pequenos para N_t.

4.2.3 Fitness Sharing

O método de *fitness sharing*, conforme detalhado na Seção 3.4.2, introduz uma modificação no algoritmo genético padrão na medida em que efetua uma redução no *fitness* de cada indivíduo por um fator proporcional ao número de indivíduos presentes na população que são similares a ele. Para isto, emprega-se uma métrica para avaliar o grau de similaridade entre indivíduos, a qual, no contexto deste trabalho, é dada pela distância Euclidiana, e também um limiar σ_s.

Este parâmetro exerce um papel importante no funcionamento do algoritmo: ele define a máxima distância permitida entre dois indivíduos para que eles ainda pertençam ao mesmo nicho (Goldberg e Richardson, 1987). No contexto de otimização de funções multimodais, o valor atribuído a σ_s deve refletir a distância aproximada entre os picos da superfície de *fitness*. Contudo, tal informação não está disponível *a priori*, o que dificulta a tarefa de estabelecer um valor para σ_s.

Além disto, conforme apontado nas Seções 2.3, 2.4 e 4.1, a depender do valor da relação

sinal-ruído e da realização dos vetores de sinal e ruído, isto é, do experimento, a superfície da função custo J_{ML} possui características bem distintas no que se refere ao número e posição dos picos, bem como na amplitude relativa entre eles. Esta particularidade do problema de estimação DOA traz novos empecilhos à missão de ajustar os parâmetros, mais especificamente, o parâmetro σ_s, do método de *fitness sharing*.

Outro parâmetro fundamental para o método de *fitness sharing* é o número de indivíduos N_P presentes na população. O valor atribuído a N_P deve ser suficiente para que o algoritmo consiga povoar os picos da superfície da função de *fitness* (Mahfoud, 1995b). Por isso, o conhecimento a respeito do número de picos existentes na superfície da função de *fitness* se faz necessário para um ajuste adequado do valor de N_P. Porém, novamente, tal informação não se encontra disponível *a priori*.

Por estes motivos, os parâmetros N_P e σ_s foram analisados através de testes de sensibilidade. Neles, foram empregados os seguintes valores para os parâmetros restantes: $p_c = 1{,}0$, $p_m = 0{,}02$, $\sigma_m = 1$. O número máximo de iterações utilizado foi $\max_{it} = 250$.

Parâmetro N_P - Número de indivíduos

Durante os testes de sensibilidade para o parâmetro N_P, atribuímos ao limiar de similaridade o valor $\sigma_s = 1,5$. Certamente este não corresponde ao valor ideal, mas, como já mencionado na Seção 4.2.2, isto não impede que tais testes evidenciem algumas diretrizes para o ajuste de N_P. Além disto, somente após a análise conjunta dos testes de sensibilidade é que serão atribuídos valores para os parâmetros.

Os resultados obtidos reforçam a mesma tendência observada para o algoritmo genético[1]: à medida que o valor de N_P aumenta, o valor RMSE obtido pelo algoritmo se aproxima da referência estabelecida pela busca em grade e a porcentagem de experimentos bem-sucedidos também se eleva. Porém, novamente temos a ressalva: o custo computacional, representado

[1]Por este motivo, optamos por omitir a apresentação das curvas de RMSE e da porcentagem de experimentos bem-sucedidos em função de N_P.

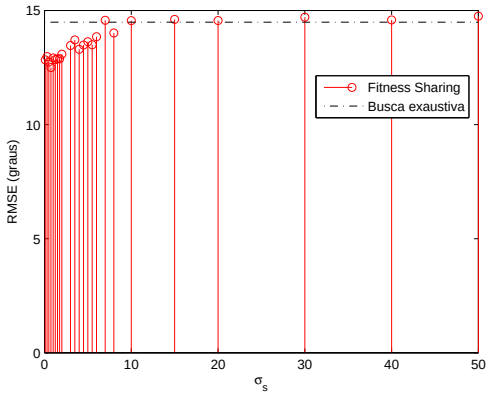
pelo número de avaliações da função de *fitness*, é uma função diretamente proporcional a N_P, como mostra a Expressão (3.11).

Parâmetro σ_s - Limiar de similaridade

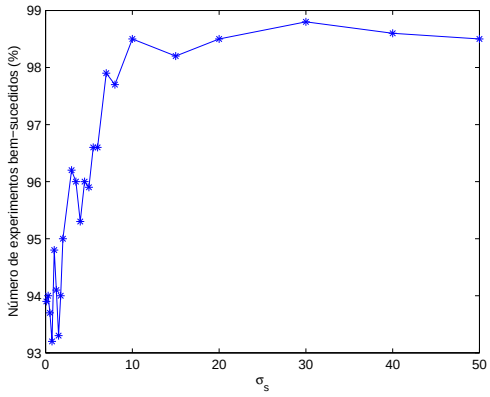
A Figura 4.3 exibe as curvas de RMSE e da porcentagem de experimentos bem-sucedidos em função do valor do parâmetro σ_s considerando o cenário e a metodologia descrita na Seção 4.2.1. Adotamos $N_P = 100$ durante os testes de sensibilidade para σ_s.

Para a SNR de -10 dB, a superfície da função custo J_{ML} apresenta um caráter multimodal mais pronunciado, como ilustrado nas Figuras 2.3(a) e 2.3(b), na Seção 2.3. Nesta situação, um valor muito pequeno de σ_s, inferior à distância média entre os picos, contribui para a convergência dos indivíduos da população em torno de poucos, ou até mesmo de um único pico. Esta perda de diversidade, por sua vez, pode ser determinante para o método de *fitness sharing* não ser capaz de localizar o ótimo global em alguns experimentos, o que, finalmente, implica na obtenção de valores RMSE distantes (para cima ou para baixo) da referência dada pela busca em grade. Podemos visualizar este fenômeno nas Figuras 4.3(a) e 4.3(b). Além disto, observamos que após um certo valor de σ_s, próximo de $\sigma_s = 10{,}0$, o valor RMSE obtido permanece bem próximo daquele associado à busca em grade, enquanto a porcentagem de experimentos bem-sucedidos está praticamente estabilizada.

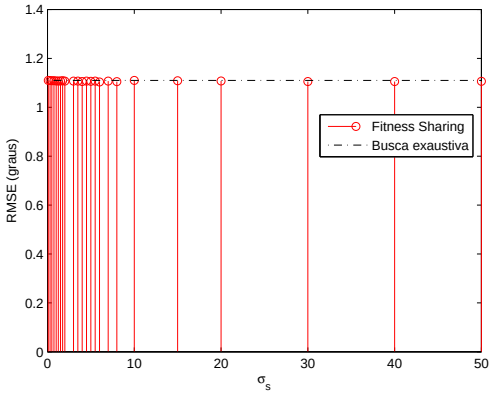
Já no caso em que a SNR é igual a 15 dB, uma vez que a superfície da função custo J_{ML} possui um lóbulo bastante pronunciado no qual estão inseridos os ótimos globais, de uma forma bastante similar à apresentada nas Figuras 2.2(a) e 2.2(b), na Seção 2.3, um valor muito elevado de σ_s pode atrapalhar a exploração local dentro dos nichos, pois o *fitness* passa a ser compartilhado com muitos indivíduos e a distinção entre os melhores indivíduos e os demais é atenuada, de forma que os ótimos perdem um pouco de sua força atratora. Com isto, ao não conseguir explorar adequadamente a região associada ao ótimo global, o método de *fitness sharing* comete um erro de estimação um pouco maior que o desejado. Este fenômeno pode ser observado na Figura 4.3(e). Neste caso, recomenda-se utilizar valores pequenos para σ_s.



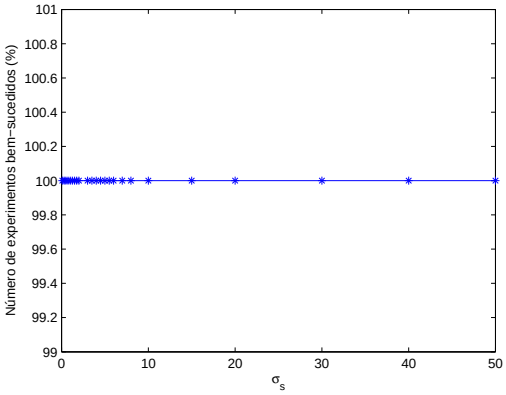
(a) SNR = −10 dB



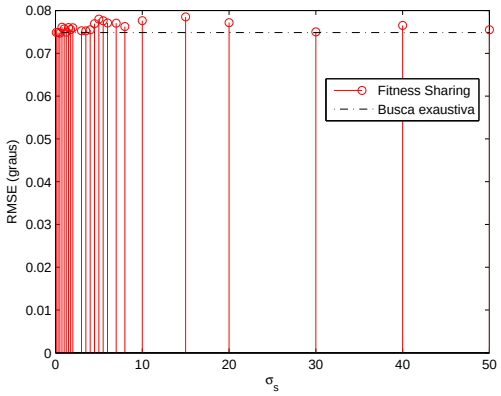
(b) SNR = −10 dB



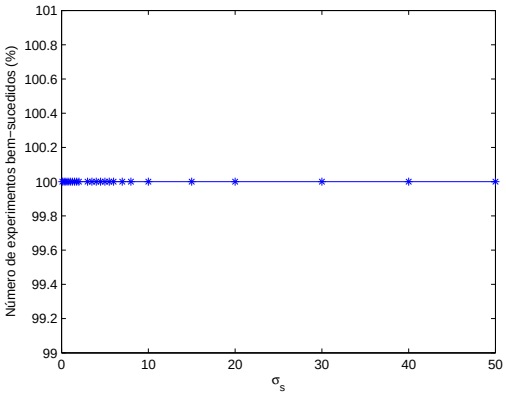
(c) SNR = −5 dB



(d) SNR = −5 dB



(e) SNR = 15 dB



(f) SNR = 15 dB

Figura 4.3: Resultados dos testes de sensibilidade para o parâmetro σ_s - *fitness sharing*.

Por fim, observamos nas Figuras 4.3(c) e 4.3(d) que o impacto de variações no valor de σ_s no desempenho do algoritmo *fitness sharing*, na situação em que a SNR é igual a -5 dB, foi praticamente desprezível.

Note que a análise realizada para os casos acima fornece duas direções opostas: por um lado, somos encorajados a aumentar o valor de σ_s com a finalidade de evitar a perda de diversidade e ampliar a exploração do espaço de busca. Por outro lado, temos a recomendação de reduzir o valor de σ_s para aprimorar a exploração local nos nichos, de modo a aumentar a precisão da busca em torno do ótimo global. É necessário, portanto, ao definir um valor para σ_s, combinar estas indicações, na tentativa de manter um desempenho adequado em ambas situações.

4.2.4 Fitness Scaling

O método de *fitness scaling*, de uma maneira semelhante ao *fitness sharing*, realiza uma modificação no valor do *fitness* por um fator proporcional ao número de indivíduos similares a cada indivíduo. Contudo, como destacado na Seção 3.4.3, com a finalidade de destacar os pontos ótimos da superfície de *fitness* em relação às demais regiões do espaço de busca, especialmente aquelas na vizinhança dos ótimos, este algoritmo aplica uma potência β_s sobre o valor do *fitness* original antes de realizar o procedimento de *sharing*.

No entanto, ao introduzir este novo parâmetro, surgem algumas questões: Qual deve ser o valor de β_s? E mais, como será a relação entre β_s e σ_s, caso ela exista? Além disso, será possível "corrigir" uma eventual escolha inadequada de um destes parâmetros por um ajuste fino do outro?

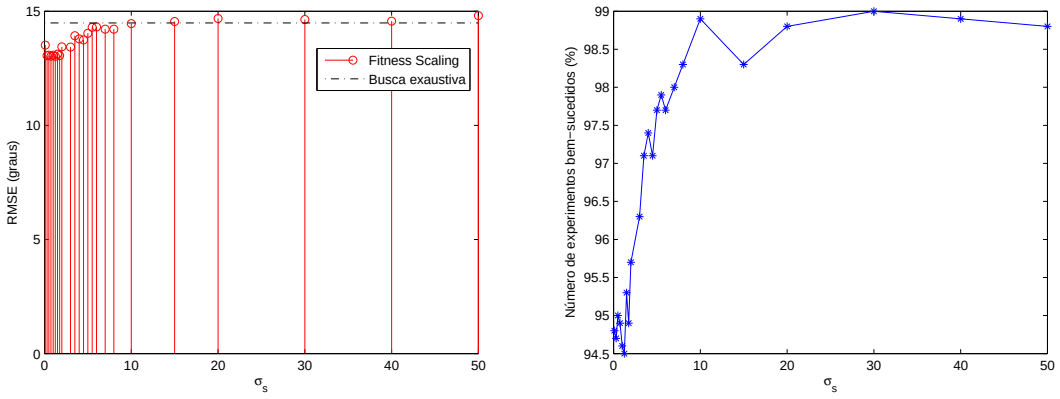
Com o propósito de respondermos a estas indagações, foram realizados testes de sensibilidade para os parâmetros N_P, σ_s e β_s. Durante estes testes, foram empregados os seguintes valores para os demais parâmetros do *fitness scaling*: $p_c = 1{,}0$, $p_m = 0{,}02$, $\sigma_m = 1$. O número máximo de iterações utilizado foi $\max_{it} = 250$.

Parâmetro N_P - Número de indivíduos

Os resultados obtidos nos testes de sensibilidade para o parâmetro N_P, usando $\sigma_s = 1,5$ e $\beta_s = 2$, fornecem a mesma indicação que aquela obtida para o algoritmo genético: à medida que o valor de N_P aumenta, o valor RMSE obtido pelo algoritmo se aproxima da referência dada pela busca em grade, enquanto a porcentagem de experimentos bem-sucedidos também se eleva. Assim sendo, julgamos não ser necessária a apresentação das curvas associadas a este caso.

Parâmetro σ_s - Limiar de similaridade

A Figura 4.4 apresenta as curvas de RMSE e do percentual de experimentos bem-sucedidos obtidas para o algoritmo *fitness scaling* em função do limiar de similaridade σ_s considerando $N_P = 100$ e $\beta_s = 2$. Optamos por apresentar os resultados somente para a SNR de -10 dB visto que, nos demais casos, o algoritmo obteve um desempenho muito próximo à referência dada pela busca em grade independentemente do valor atribuído ao parâmetro σ_s.



(a) SNR = -10 dB (b) SNR = -10 dB

Figura 4.4: Resultados dos testes de sensibilidade para o parâmetro σ_s - *fitness scaling*.

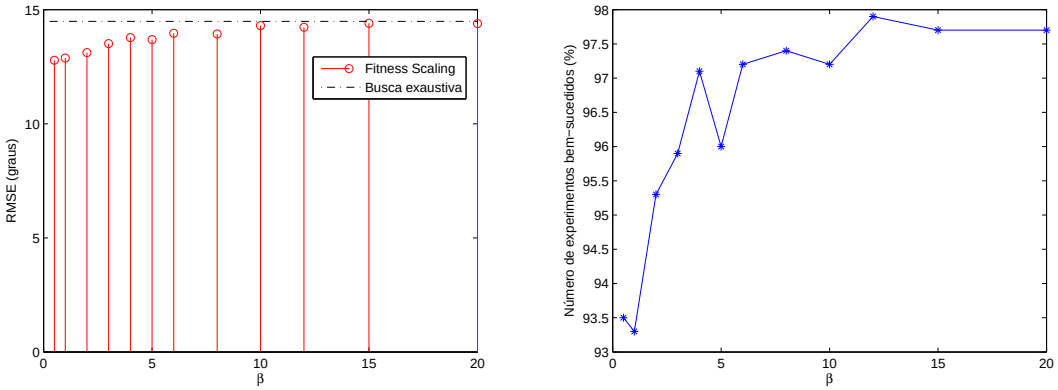
As curvas exibidas na Figura 4.4 apresentam um comportamento bastante similar ao verificado para o algoritmo *fitness sharing*. Podemos notar que um valor muito pequeno de σ_s se mostra inadequado uma vez que, nesta configuração, foram obtidos valores RMSE inferiores ao

da busca em grade, o que significa que o método de *fitness sharing* não foi capaz de localizar o ótimo global em alguns experimentos.

Como o valor atribuído a β_s durante os testes de sensibilidade não é muito diferente de $\beta_s = 1$, o qual caracteriza o algoritmo *fitness sharing*, era de se esperar que, assim como este último método, após um certo valor de σ_s, fossem obtidos valores RMSE próximos daquele associado à busca em grade. Isto realmente se comprovou, conforme mostra a Figura 4.4(a). E, assim como ocorreu para o *fitness sharing*, a porcentagem de experimentos bem-sucedidos pouco se altera quando σ_s é suficientemente elevado.

Parâmetro β_s - Fator de escalonamento

Apresentamos na Figura 4.5 as curvas de RMSE e do percentual de experimentos bem-sucedidos em função do valor do parâmetro β_s para a SNR de -10 dB considerando $N_P = 100$ e $\sigma_s = 1{,}5$. Nos casos em que SNR $= 15$ dB e SNR $= -5$ dB, foi verificado que o algoritmo *fitness scaling* consegue estimar adequadamente os ângulos de chegada para todos os valores atribuídos a β_s, de forma que os valores RMSE obtidos permaneceram muito próximos ao valor associado à busca em grade. Por isso, as respectivas curvas não são exibidas.



(a) SNR $= -10$ dB (b) SNR $= -10$ dB

Figura 4.5: Resultados dos testes de sensibilidade para o parâmetro β_s - *fitness scaling*.

Podemos notar que, à medida que o valor de β_s aumenta, o desempenho do algoritmo *fitness*

scaling, expresso em termos de RMSE e do percentual de experimentos bem-sucedidos, melhora. Isto nos leva à conclusão que é conveniente utilizar valores elevados para a potência β_s.

Contudo, quando juntamos as Figuras 4.5 e 4.4, podemos perceber uma interessante relação entre os parâmetros β_s e σ_s:

1. embora a Figura 4.4 indique que um valor pequeno de σ_s não é adequado, foi possível obter um bom desempenho neste caso ao empregar um valor elevado para o parâmetro β_s, como mostra a Figura 4.5.

2. embora o uso de um valor pequeno para β_s não seja recomendado, como indica a Figura 4.5, foi possível atingir um bom desempenho nesta situação ao empregar um valor elevado para σ_s, conforme ilustra a Figura 4.4.

Isto confere ao método de *fitness scaling* maior robustez se comparado ao *fitness sharing*, pois a tarefa de ajustar os parâmetros, embora ainda exija cuidados, é facilitada pela possibilidade de combinar o efeito dos parâmetros β_s e σ_s com a finalidade de obter um melhor desempenho.

4.2.5 Clearing

Em vez de realizar um compartilhamento, o método de *clearing* preserva inalterado o *fitness* dos κ melhores indivíduos presentes em cada nicho, ao passo que zera o *fitness* dos demais indivíduos. Os principais parâmetros deste algoritmo são: o número N_P de indivíduos presentes na população, o limiar de similaridade σ_s e a capacidade dos nichos κ. Por isso, foram realizados testes de sensibilidade envolvendo estes três parâmetros. Os demais foram ajustados da seguinte forma: $p_c = 1{,}0$, $p_m = 0{,}03$, $\sigma_m = 1$ e $\max_{it} = 250$.

Parâmetro N_P - Número de indivíduos

Os resultados obtidos nos testes de sensibilidade para o parâmetro N_P, usando $\sigma_s = 1{,}5$ e $\kappa = 1$, também indicam um progresso de desempenho do método de *clearing* à medida que o

valor de N_P aumenta, assim como fora verificado para os algoritmos genético, *fitness sharing* e *fitness scaling*.

Parâmetro σ_s - Limiar de similaridade

Utilizando $N_P = 100$ e $\kappa = 1$, foram realizados testes de sensibilidade para o parâmetro σ_s. Foi observado um comportamento bastante similar ao verificado para o algoritmo *fitness sharing*: 1) quando σ_s é pequeno, especialmente no caso em que a SNR é igual a -10 dB, foram obtidos valores RMSE inferiores ao da busca em grade. 2) após um certo valor, o desempenho do algoritmo tende à referência dada pela busca em grade e o parâmetro σ_s passa a não interferir significativamente.

Parâmetro κ - Capacidade dos nichos

Os resultados obtidos para SNR = 15 dB e SNR = -5 dB, considerando $N_P = 100$ e $\sigma_s = 1{,}5$, mostram que o método de *clearing* foi capaz de determinar a posição do ótimo global da função custo J_{ML} em todos os experimentos, independentemente do valor atribuído a κ. Porém, no caso em que SNR = -10 dB, observamos uma suave degradação de desempenho, particularmente na porcentagem de experimentos bem-sucedidos, à medida que o valor de κ aumenta.

Intuitivamente, esta tendência observada nos testes de sensibilidade pode ser compreendida da seguinte forma: ao permitir um número maior de indivíduos por nicho, aumentamos a chance de ocorrerem aglomerações em poucos nichos. Isto, por sua vez, reduz as possibilidades de se explorar outros nichos ecológicos, pois a busca fica concentrada nos nichos já localizados. Desta forma, é possível que a região onde se encontra o ótimo global não seja visitada, fazendo com que o algoritmo falhe na estimação dos ângulos de chegada. Por isso, fica a indicação de que pode ser vantajoso utilizar valores pequenos para κ.

4.2.6 Estratégias Evolutivas

As estratégias evolutivas apresentadas na Seção 3.4.5 requerem a definição dos seguintes parâmetros: μ, λ, β_ϑ, $\dot{\tau}$ e τ. Os dois primeiros estão associados à dinâmica da população, definindo, respectivamente, o número de indivíduos presentes na população e o número de descendentes gerados em cada iteração. Os demais representam constantes empregadas durante a etapa de mutação dos parâmetros de controle.

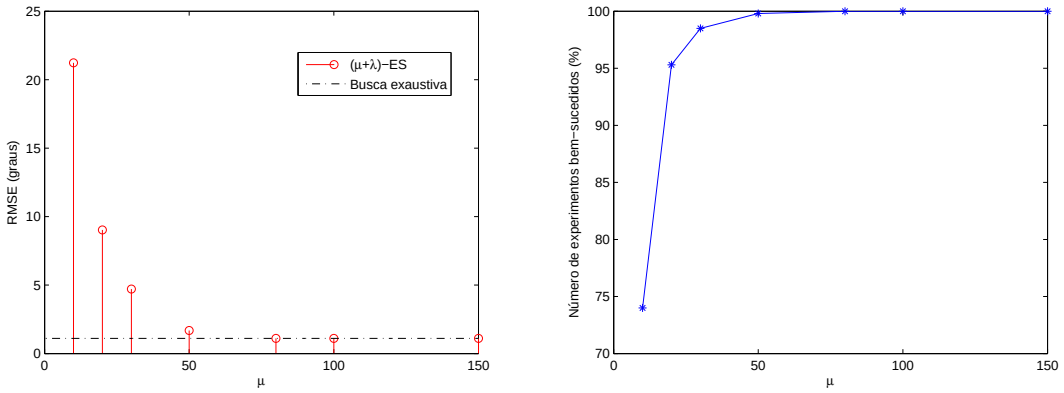
Por isto, decidimos realizar testes de sensibilidade somente para os parâmetros μ e λ. As constantes de mutação foram definidas segundo (Schwefel, 1995): $\beta_\vartheta = 5\pi/180$, $\dot{\tau} = (2M)^{\frac{1}{2}}$ e $\tau = (4M)^{\frac{1}{4}}$, onde M é o número de ângulos de chegada.

Neste trabalho, como já destacado na Seção 3.4.5, foram utilizadas duas estratégias evolutivas para a estimação dos ângulos de chegada: $(\mu + \lambda)$-ES e (μ, λ)-ES. Estas propostas consagradas diferem apenas no repertório de indivíduos utilizado na etapa de seleção: a primeira utiliza tanto a população de pais quanto a dos descendentes; a segunda, por outro lado, somente a população de descendentes. Por isso, espera-se que o impacto dos parâmetros μ e λ seja distinto. Logo, os dois casos serão analisados separadamente.

$(\mu + \lambda)$-ES

Primeiramente, foram realizados testes de sensibilidade para o parâmetro μ, nos quais empregamos $\lambda = 20$. O critério de parada adotado foi um número máximo de iterações igual a $\max_{it} = 300$. A Figura 4.6 apresenta os resultados obtidos somente para o caso em que a relação sinal-ruído é igual a -5 dB, uma vez que nos outros casos, o mesmo comportamento foi verificado.

Podemos notar que, quando μ é pequeno, os valores RMSE obtidos para a estratégia evolutiva estão bem distantes do valor associado à busca em grade. Porém, conforme μ aumenta, temos um progresso de desempenho, até o momento em que o algoritmo consegue atingir um desempenho praticamente similar ao da busca em grade. Ou seja, uma vez definido o número

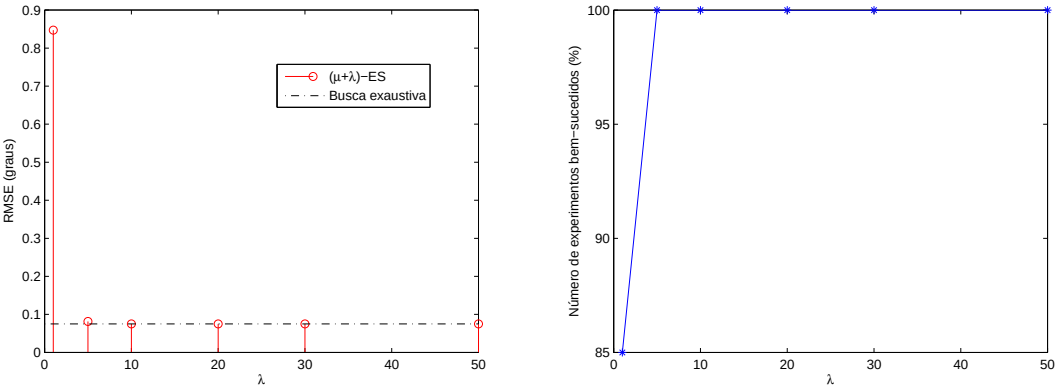


(a) SNR = −5 dB (b) SNR = −5 dB

Figura 4.6: Resultados dos testes de sensibilidade para o parâmetro μ - $(\mu + \lambda)$-ES.

de descendentes gerados por iteração, quanto maior for o tamanho da população, mais chances o algoritmo terá de gerar novos indivíduos de boa qualidade, afinal, há uma disponibilidade maior de material genético para ser utilizado na busca. Isto contribui, finalmente, para um melhor desempenho do algoritmo.

A Figura 4.7, por sua vez, apresenta os resultados obtidos nos testes de sensibilidade realizados para o parâmetro λ considerando $\mu = 100$ apenas no caso em que SNR = 15 dB, já que o mesmo é suficiente para explicitar a influência deste parâmetro no desempenho do algoritmo.



(a) SNR = 15 dB (b) SNR = 15 dB

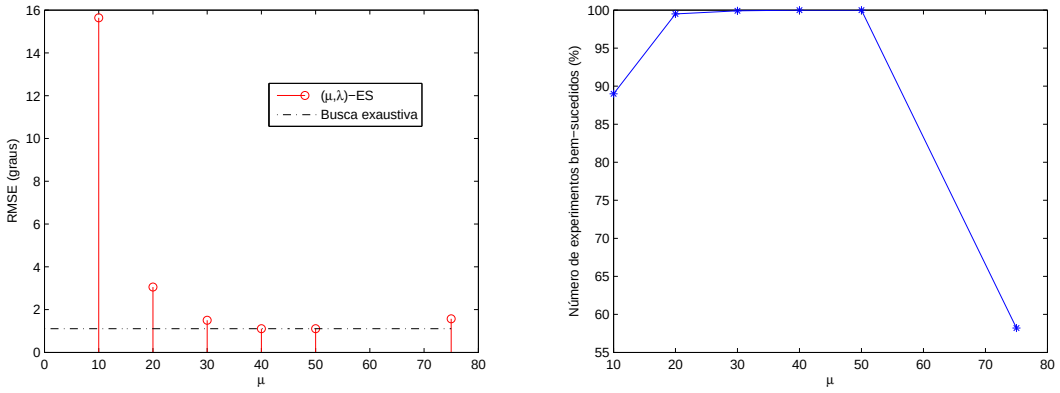
Figura 4.7: Resultados dos testes de sensibilidade para o parâmetro λ - $(\mu + \lambda)$-ES.

É possível concluir, através da análise das curvas mostradas na Figura 4.7, que a partir de

um certo valor de λ, o algoritmo passa a atingir um desempenho muito próximo ao associado à busca em grade. Quando λ é pequeno, poucos descendentes são gerados a partir da população, de forma que, a cada iteração, pouca novidade é introduzida. Logo, o processo de busca realizado pelo algoritmo se torna bastante lento e, dentro do limite de iterações estabelecido, pode não ser capaz de localizar o ótimo global. Por isso, é conveniente aumentar o tamanho da população de descendentes.

(μ, λ)-ES

A influência do parâmetro μ no desempenho da estratégia evolutiva (μ, λ)-ES foi estudada através de testes de sensibilidade. Para isto, atribuímos ao número de descendentes gerados por iteração o valor $\lambda = 100$, além de estabelecermos um número máximo de iterações igual a $\max_{it} = 200$. A Figura 4.8 apresenta os resultados obtidos somente para o caso em que a relação sinal-ruído é igual a -5 dB, pois o mesmo comportamento foi verificado para os outros valores de SNR.



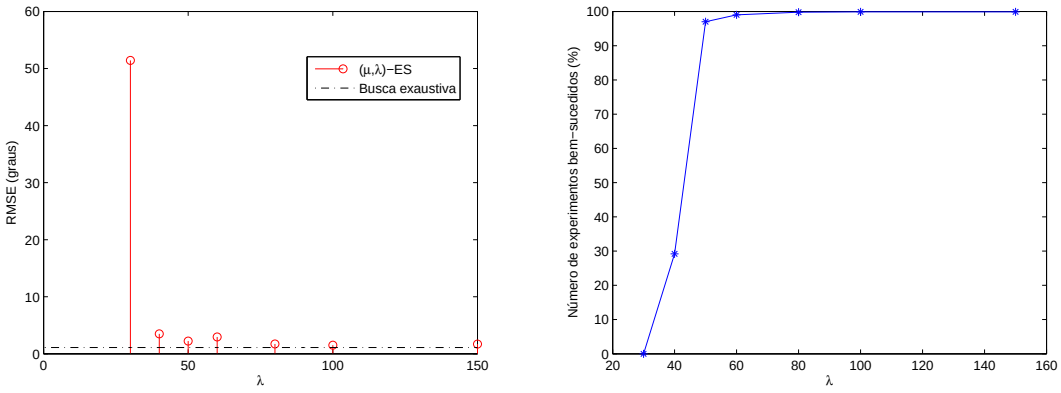
(a) SNR $= -5$ dB (b) SNR $= -5$ dB

Figura 4.8: Resultados dos testes de sensibilidade para o parâmetro μ - (μ, λ)-ES.

É possível notar na Figura 4.8 que, a partir de um certo valor de μ, o desempenho do algoritmo se aproxima do valor RMSE associado à busca em grade. Contudo, quando μ é muito grande, ocorrem falhas na estimação dos ângulos de chegada, pois, neste caso, como

o valor de μ está muito próximo de λ, quase não temos pressão seletiva, pois praticamente todos os descendentes gerados são selecionados para a próxima iteração. Ou seja, o sucesso do algoritmo na otimização da função custo J_{ML} depende exclusivamente das operações de mutação e *crossover*, e não do operador de seleção, o que caminha na direção contrária ao espírito dos algoritmos evolutivos.

Por fim, apresentamos na Figura 4.9 as curvas de RMSE e do percentual de experimentos bem-sucedidos em função do parâmetro λ considerando $\mu = 20$ e, novamente, somente a SNR de -5 dB.



(a) SNR $= -5$ dB (b) SNR $= -5$ dB

Figura 4.9: Resultados dos testes de sensibilidade para o parâmetro λ - (μ, λ)-ES.

Podemos concluir, a partir da Figura 4.9, que o valor do parâmetro λ deve ser suficientemente elevado para que o algoritmo atinja um bom desempenho na estimação dos ângulos de chegada. Percebemos que, enquanto λ é pequeno, poucos descendentes são gerados a partir de uma população com $\mu \leq \lambda$ indivíduos. Esta situação deve ser evitada, pois novamente o operador de seleção se torna praticamente inoperante, deixando assim o algoritmo à mercê de uma mutação ou recombinação bem-sucedida para encontrar o ótimo global.

4.2.7 Evolução Diferencial

O algoritmo denominado Evolução Diferencial, cuja descrição encontra-se na Seção 3.4.6, requer o ajuste de apenas três parâmetros: o número de indivíduos da população (N_P), a constante de *crossover* (CR) e o passo da mutação (F). Embora existam algumas recomendações de valores para os dois últimos, preferimos realizar testes de sensibilidade considerando os três parâmetros mencionados acima a fim de obter uma visão mais ampla acerca da interferência dos mesmos no desempenho do algoritmo, além de verificar se tais recomendações efetivamente são adequadas para este problema em particular.

Parâmetro N_P - Número de indivíduos

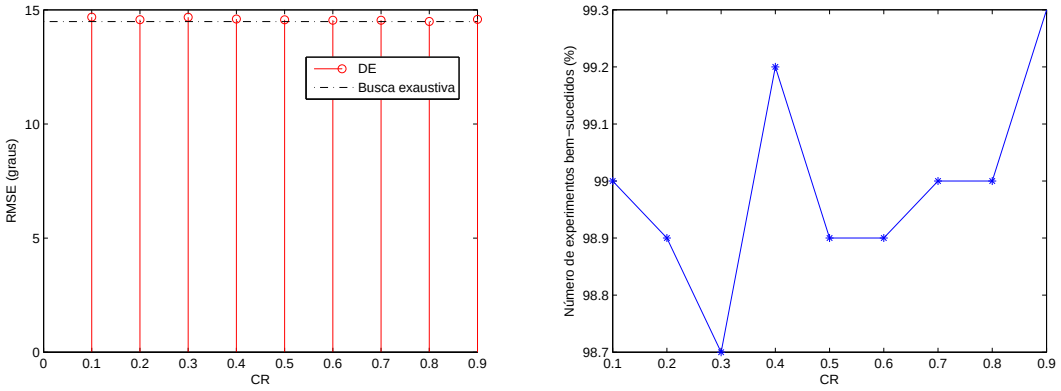
Os testes de sensibilidade relacionados ao parâmetro N_P foram realizados segundo a mesma metodologia exposta na Seção 4.2.1. Os valores atribuídos aos demais parâmetros foram: $F = 0{,}5$, $CR = 0{,}9$ e $\max_{it} = 200$. Os resultados obtidos se assemelham àqueles associados ao algoritmo genético. Ou seja, foi observado que, conforme o valor de N_P aumenta, o algoritmo DE consegue estimar os ângulos de chegada com melhor precisão, além de obter êxito em uma porcentagem cada vez maior de experimentos.

Convém ressaltar que o valor de N_P influencia o desempenho do algoritmo de uma forma um pouco diferente dependendo do valor da relação sinal-ruído: no caso em que a SNR é igual a 15 dB, não é necessário um aumento significativo no valor de N_P para que o desempenho deste algoritmo tenda à referência dada pela busca em grade; porém, para a SNR de -10 dB, observamos que o aumento em N_P precisa ser mais acentuado. Isto se deve à natureza distinta das superfícies a serem otimizadas: como destacado na Seção 2.3, à medida que o valor da SNR diminui, a superfície da função custo J_{ML} se torna cada vez mais multimodal e variante de experimento para experimento. Logo, é de se esperar que a exigência em termos de número de amostras do espaço de busca, que por sua vez está relacionado ao número de indivíduos presentes na população, seja maior nos caso mais críticos, os quais estão relacionados às SNRs

baixas.

Parâmetro CR - Constante de crossover

A Figura 4.10 apresenta as curvas de RMSE e da porcentagem de experimentos bem-sucedidos obtidas nos testes de sensibilidade envolvendo o parâmetro $CR \in [0, 1]$. Para sua realização, foram adotados os seguintes valores para os demais parâmetros: $N_P = 100$, $F = 0{,}5$ e $\max_{it} = 200$. Para as SNRs de 15 dB e -5 dB, o algoritmo foi capaz de estimar corretamente os ângulos de chegada, obtendo um desempenho similar ao da busca em grade, não sendo afetado pelo valor de CR. Por isso, apresentamos somente as curvas para o caso em que a SNR é igual a -10 dB.



(a) SNR = -10 dB (b) SNR = -10 dB

Figura 4.10: Resultados dos testes de sensibilidade para o parâmetro CR - Evolução Diferencial.

Podemos observar na Figura 4.10 uma ligeira vantagem quando o parâmetro CR assume o valor 0,9. Entretanto, o desempenho nos demais casos é muito similar em termos dos valores RMSE, embora com uma porcentagem de experimentos bem-sucedidos um pouco inferior. Isto mostra que, como um todo, a influência do parâmetro CR no desempenho final do algoritmo DE é até certo ponto reduzida.

Parâmetro F - Passo da mutação

Os resultados obtidos nos testes de sensibilidade para o parâmetro $F \in [0, 2]$, usando $N_P = 100$, $CR = 0{,}9$ e $\max_{it} = 200$, indicam que: 1) para as SNRs de 15 dB e -5 dB, DE obtém um desempenho praticamente equivalente ao da busca em grade independentemente do valor de F. 2) para a SNR de -10 dB, o algoritmo DE atinge o melhor desempenho quando o valor de F está próximo de 0,5 ou próximo de 1,3. No entanto, assim como observado nos testes de sensibilidade referentes ao parâmetro CR, o impacto do valor do parâmetro F também se mostra reduzido.

4.2.8 CLONALG

O algoritmo CLONALG se inspira nos princípios de seleção clonal e maturação de afinidade, utilizados para explicar a resposta adaptativa do sistema imunológico natural, para manipular uma população de soluções candidatas com o propósito de resolver problemas, inclusive de otimização, como o problema de estimação DOA. Para isto, é necessário atribuir valores adequados para os seguintes parâmetros: N_P, N_c, ρ, T_{Ab} e p_{Ab}, além do número máximo de iterações $\max_{it}$.

Dentre eles, foram selecionados os seguintes parâmetros para uma análise mais detalhada mediante testes de sensibilidade: N_P, N_c e ρ. Os demais foram ajustados considerando um conjunto de experimentos preliminares e receberam os valores $T_{Ab} = 20$, $p_{Ab} = 0{,}15$ e $\max_{it} = 250$. A seguir, apresentamos os principais resultados obtidos nos testes de sensibilidade.

Parâmetro N_P - Número de indivíduos

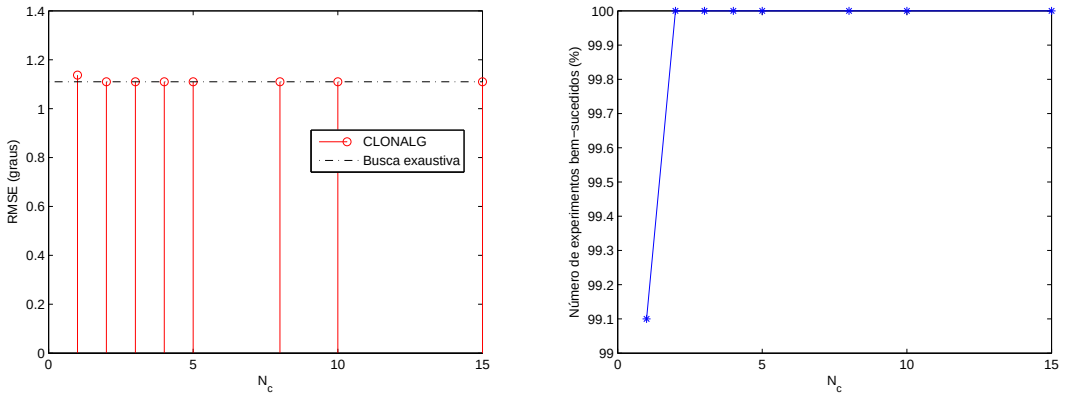
Utilizando $N_c = 10$ e $\rho = 2$, foram realizados testes de sensibilidade para o parâmetro N_P, e os resultados observados confirmam a tendência verificada para os algoritmos anteriores: conforme o valor de N_P aumenta, obtém-se um progresso no desempenho do algoritmo, pois o erro de estimação, expresso através da medida RMSE, se aproxima da referência estabelecida

pela busca em grade, e o percentual de experimentos bem-sucedidos aumenta.

Parâmetro N_c - Número de clones

Intuitivamente, podemos deduzir que, quanto maior o número de clones gerados para cada anticorpo durante a etapa de seleção clonal e expansão, maiores são as chances de se explorar adequada e rapidamente as regiões do espaço de busca povoadas pelos anticorpos. Por outro lado, se o valor de N_c for muito pequeno, poucas são as mutações eficazes, isto é, que produzem clones mutados melhores que o anticorpo original, de modo que o progresso do algoritmo pode se tornar muito lento.

A Figura 4.11 exibe os resultados obtidos nos testes de sensibilidade para o parâmetro N_c, sendo $N_P = 50$ e $\rho = 2$, considerando o caso em que a SNR é igual a -5 dB. Observe que as curvas obtidas confirmam a expectativa descrita anteriormente com respeito ao impacto do valor de N_c no comportamento e desempenho do algoritmo CLONALG.



(a) SNR = -5 dB (b) SNR = -5 dB

Figura 4.11: Resultados dos testes de sensibilidade para o parâmetro N_c - CLONALG.

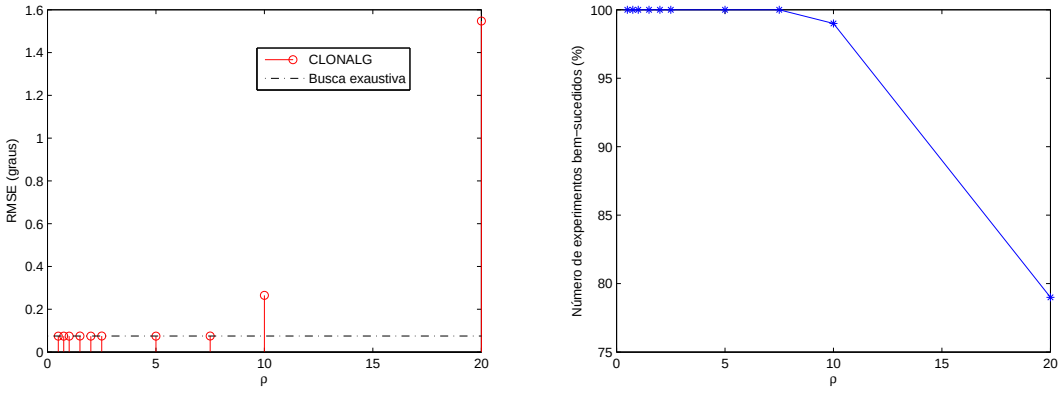
Parâmetro ρ - Fator de decaimento da mutação

Seguindo um raciocínio semelhante ao realizado para o parâmetro N_c, podemos inferir qual deve ser a influência do parâmetro ρ, cuja função é regular a abertura da mutação gaussiana

efetuada sobre os clones, conforme mostra a Expressão (3.22), no desempenho do algoritmo CLONALG:

1. se o valor de ρ for muito pequeno, isto significa que, potencialmente, a mutação sofrida pelos clones será bastante acentuada, dado o valor elevado do desvio padrão da variável gaussiana. Isto pode ser indesejável, pois, desta forma, praticamente desaparece a relação de semelhança entre o anticorpo original e seus clones, de modo que a idéia de uma exploração local na região ocupada pelo anticorpo pode ficar comprometida.

2. por outro lado, se o valor de ρ for muito elevado, os clones sofrem mutações muito sutis, de modo que pouco diferem do anticorpo original. Isto pode tornar o progresso do algoritmo muito lento. Além disto, dependendo das características da superfície da função sendo otimizada, a inicialização da população, assim como as inserções de novos anticorpos realizadas em alguns momentos, passam a exercer uma maior influência no comportamento do algoritmo, já que as mutações contribuem apenas suave e lentamente para a exploração do espaço de busca.



(a) SNR = 15 dB (b) SNR = 15 dB

Figura 4.12: Resultados dos testes de sensibilidade para o parâmetro ρ - CLONALG.

Os resultados obtidos nos testes de sensilidade realizados para o parâmetro ρ, empregando $N_P = 50$ e $N_c = 10$, estão de acordo com as observações previamente apontadas, como mostra a Figura 4.12, considerando o caso em que a relação sinal-ruído é igual a 15 dB.

4.2.9 Opt-aiNet

O algoritmo denominado *opt-aiNet* representa uma extensão do CLONALG na medida em que incorpora elementos da teoria da rede imunológica através da introdução de mecanismos de supressão e inserção de indivíduos baseados em critérios de estagnação da população de anticorpos. Assim como as técnicas de *niching*, abordadas nas Seções 3.4.2 a 3.4.4, o conceito de similaridade é definido com base na distância entre os anticorpos no espaço real, de modo que dois anticorpos são considerados similares quando a distância entre eles for inferior a um limiar σ_s.

Uma vez que a rede *opt-aiNet* é capaz de dinamicamente regular o número de anticorpos presentes na população, o parâmetro N_P, antes empregado para definir o tamanho da população de soluções candidatas, agora estabelece apenas o número de anticorpos presentes na população inicial. Outra novidade em relação ao CLONALG é o estabelecimento de uma condição baseada na estagnação da população de anticorpos para a realização das operações de inserção e supressão. Esta etapa exige a definição dos parâmetros T_{Ab}, p_{Ab} e F_{dif}, como discutido na Seção 3.5.2. Os demais parâmetros utilizados pela *opt-aiNet* são os mesmos do CLONALG, a saber, N_c e ρ, além do número máximo de iterações permitidas $\max_{it}$.

Dentre todos os parâmetros, dedicamos maior atenção ao ajuste e análise de N_c, ρ e σ_s por meio de testes de sensibilidade. Os parâmetros restantes foram avaliados com base em um conjunto de experimentos preliminares, sendo por fim definidos da seguinte forma: $N_P = 40$, $T_{Ab} = 10$, $F_{dif} = 0{,}005$, $p_{Ab} = 0{,}2$ e $\max_{it} = 300$. A seguir, apresentamos os principais observações extraídas a partir dos testes de sensibilidade.

Parâmetro N_c - Número de clones

Fixando $\rho = 2$ e $\sigma_s = 1{,}4$, foram realizados testes de sensibilidade envolvendo o parâmetro N_c, segundo a metodologia apresentada na Seção 4.2.1. A partir dos resultados obtidos, foi possível observar que, nos casos em que SNR $= -10$ dB e SNR $= -5$ dB, o desempenho da rede *opt-aiNet* ao estimar os ângulos de chegada se mostrou adequado, já que os valores RMSE obtidos estão muito próximos ao associado à busca em grade, e praticamente não houve variações para todos os valores atribuídos a N_c.

Apesar disto, para uma SNR de 15 dB, verificou-se que, quando N_c é muito pequeno, embora a rede *opt-aiNet* tenha sido capaz de localizar o ótimo global em todos os experimentos, houve uma perda de precisão nas estimativas alcançadas, de modo que o valor RMSE obtido pela *opt-aiNet*, ainda que próximo, não atingiu a referência de desempenho determinada pela busca em grade. No entanto, com o aumento de N_c, tais imprecisões são atenuadas, fazendo com que o desempenho da *opt-aiNet* seja similar ao da busca em grade.

Parâmetro ρ - Fator de decaimento da mutação

Embora as observações apontadas na Seção 4.2.8 com relação ao parâmetro ρ também vigorem para a rede *opt-aiNet*, os resultados obtidos nos testes de sensibilidade para este parâmetro, usando $N_c = 10$ e $\sigma_s = 1{,}4$, indicam que o desempenho da *opt-aiNet* praticamente não é afetado por variações no valor de ρ, embora uma suave degradação tenha sido verificada quando ρ assume valores elevados, no caso em que a relação sinal-ruído é igual a 15 dB.

Mas, qual será a razão desta mudança? Por que o valor atribuído a ρ afeta de modo mais acentuado o desempenho do algoritmo CLONALG? Apontamos a seguir dois fatores que explicam esta questão:

1. O algoritmo CLONALG possui um viés elitista, pois os clones gerados e mutados substituem os anticorpos originais somente quando superam estes últimos em

termos de *fitness*. A rede *opt-aiNet* também possui esta característica, mas, ao introduzir um mecanismo de supressão de anticorpos similares, ela tende a evitar a ocorrência de aglomerações em torno de regiões já ocupadas, estimulando assim a exploração de novas regiões do espaço de busca, de forma que as chances de se encontrar o ótimo global tendem a ser maiores.

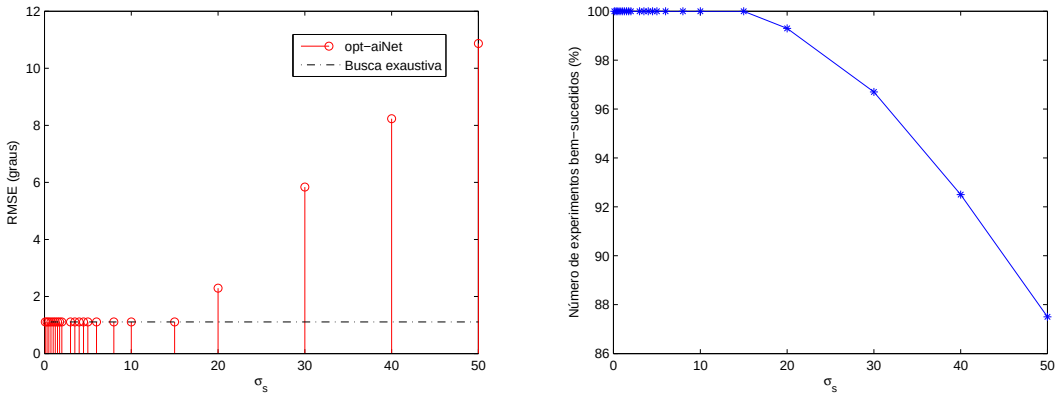
2. O algoritmo CLONALG, em períodos de T_{Ab} iterações, realiza a substituição de uma porcentagem p_{Ab} dos piores anticorpos por outros gerados aleatoriamente. Podemos interpretar esta operação como uma supressão baseada em *fitness*. Por outro lado, a rede *opt-aiNet*, ao empregar um mecanismo de supressão que, em vez de considerar o *fitness* dos anticorpos, baseia-se na similaridade existente entre eles, como descrito na Seção 3.5.2, tende a evitar a perda daqueles anticorpos que, embora ainda não possuam valores elevados de *fitness*, estejam explorando regiões promissoras do espaço de busca.

Os aspectos mencionados acima ressaltam algumas características inerentes à *opt-aiNet* que a tornam menos dependente do parâmetro ρ que o algoritmo CLONALG, o que, de fato, foi verificado nos testes de sensibilidade.

Parâmetro σ_s - Limiar de similaridade

A Figura 4.13 exibe as curvas de RMSE e do percentual de experimentos bem-sucedidos em função do parâmetro σ_s obtidas como resultado dos testes de sensibilidade, considerando $N_c = 10$ e $\rho = 2$, para o caso em que a SNR é igual a -5 dB. Enfatizamos que os outros casos foram omitidos, uma vez que o mesmo comportamento aqui mostrado foi verificado.

É possível notar na Figura 4.13 que, para valores elevados de σ_s, a rede *opt-aiNet* sofre uma degradação de desempenho. É preciso lembrar que o fato de o valor de σ_s ser elevado faz com que anticorpos cada vez mais distantes no espaço de busca sejam considerados similares e, conseqüentemente, durante a etapa de supressão, eliminados da população.



(a) SNR = −5 dB (b) SNR = −5 dB

Figura 4.13: Resultados dos testes de sensibilidade para o parâmetro σ_s - *Opt-aiNet*.

Isto pode ser inconveniente, uma vez que diferentes picos do espaço de busca, ocupados por diferentes anticorpos, passam a ser vistos como uma única região, similar a um nicho, que continuará sendo explorada apenas pelo melhor entre esses anticorpos. Contudo, tal anticorpo pode não estar situado no melhor dos picos existentes nesta região, de modo que este último não será explorado. Desta maneira, as chances de a rede *opt-aiNet* não encontrar o ótimo global aumentam, o que explica a degradação de desempenho verificada nos resultados dos testes de sensibilidade nestas condições.

4.2.10 Particle Swarm

O algoritmo *Particle Swarm*, conforme detalhado na Seção 3.6.1, emprega os princípios de uma teoria sócio-cognitiva, a qual modela a forma como as sociedades humanas processam o conhecimento, para manipular e modificar uma população de soluções candidatas (partículas) com a finalidade de resolver problemas de otimização, como o problema de estimação DOA, segundo o critério ML.

Para seu funcionamento, o PS requer a definição dos seguintes parâmetros : N_P, ν_{min}, ν_{max}, L_1 e L_2. Dentre eles, somente N_P e ν_{max} foram analisados por meio de testes de sensibilidade. As constantes de aceleração L_1 e L_2, as quais participam da etapa de atualização do vetor

velocidade de cada partícula, foram ajustadas e seus valores são $L_1 = L_2 = 2{,}05$, obedecendo assim a recomendação dada por Kennedy (Kennedy, 2004). Já o parâmetro ν_{min}, por questões de simplicidade, foi fixado com o valor oposto a ν_{max}, isto é, $\nu_{min} = -\nu_{max}$. Por fim, estabelecemos $\max_{it} = 350$. A seguir, apresentamos os principais resultados obtidos nos testes de sensibilidade.

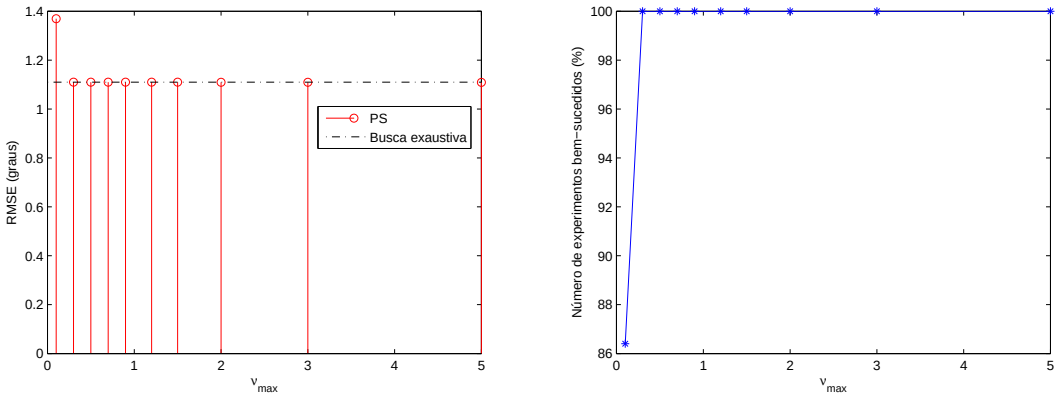
Parâmetro N_P - Número de partículas

Os testes de sensibilidade relacionados ao parâmetro N_P foram realizados de acordo com a metodologia apresentada na Seção 4.2.1. Nestes testes, foi empregado $\nu_{max} = 0{,}6$. Os resultados obtidos se assemelham àqueles associados aos demais algoritmos de computação natural: novamente, foi observado que, conforme o valor de N_P aumenta, obtém-se um progresso de desempenho, de modo que os valores RMSE associados ao algoritmo PS se aproximam da referência dada pela busca em grade, enquanto a porcentagem de experimentos bem-sucedidos aumenta.

Parâmetro ν_{max} - Limite do vetor velocidade

A Figura 4.14 apresenta as curvas de RMSE e do percentual de experimentos bem-sucedidos em função do parâmetro ν_{max} obtidas como resultado dos testes de sensibilidade, considerando $N_P = 100$ e o caso em que a SNR é igual a -5 dB. Os demais casos foram omitidos uma vez que os respectivos resultados fornecem as mesmas indicações que as extraídas no caso aqui exibido.

Podemos observar na Figura 4.14 que o algoritmo *particle swarm* alcança um desempenho praticamente equivalente à busca em grade para praticamente todos os valores de ν_{max}, exceto quando ν_{max} é muito pequeno. Nesta situação, em cada iteração, cada partícula sofre um deslocamento muito pequeno, de forma que o progresso do algoritmo pode se tornar muito lento, o que explica a degradação de desempenho observada na Figura 4.14.



(a) SNR = −5 dB (b) SNR = −5 dB

Figura 4.14: Resultados dos testes de sensibilidade para o parâmetro ν_{max} - *Particle Swarm*.

4.2.11 Particle Swarm com fator de inércia

A introdução do fator de inércia ζ, proposta em (Shi e Eberhart, 1999), visa balancear as parcelas associadas à busca local e global existentes na Expressão (3.26) a fim de tornar o algoritmo *particle swarm* melhor habilitado na tarefa de encontrar o ótimo global de funções.

Os principais parâmetros desta nova versão do *particle swarm*, a saber, N_P, ζ e ν_{max}, também foram analisados por meio de testes de sensibilidade. As constantes de aceleração L_1 e L_2, bem como o parâmetro ν_{min}, receberam os mesmos valores apresentados na Seção 4.2.10. Por outro lado, o número máximo de iterações $\max_{it}$ empregado durante os testes de sensibilidade foi igual a 500.

Parâmetro N_P - Número de partículas

Utilizando $\nu_{max} = 1{,}5$ e $\zeta = 1{,}1$, foram realizados testes de sensibilidade para o parâmetro N_P e os resultados observados confirmam a tendência verificada para a versão original do *particle swarm*: conforme o valor de N_P aumenta, observa-se um progresso no desempenho do algoritmo, já que o erro de estimação, expresso através da medida RMSE, se aproxima da referência estabelecida pela busca em grade, e o percentual de experimentos bem-sucedidos aumenta.

Parâmetro ν_{max} - Limite do velor velocidade

Os resultados obtidos nos testes de sensibilidade para o parâmetro ν_{max}, considerando $N_P = 100$ e $\zeta = 1{,}1$, exibem o mesmo comportamento que aquele apresentado anteriormente para o PS em sua versão original: exceto quando o valor de ν_{max} é muito pequeno, o algoritmo consegue atingir praticamente o mesmo desempenho que a busca em grade.

Parâmetro ζ - Fator de inércia

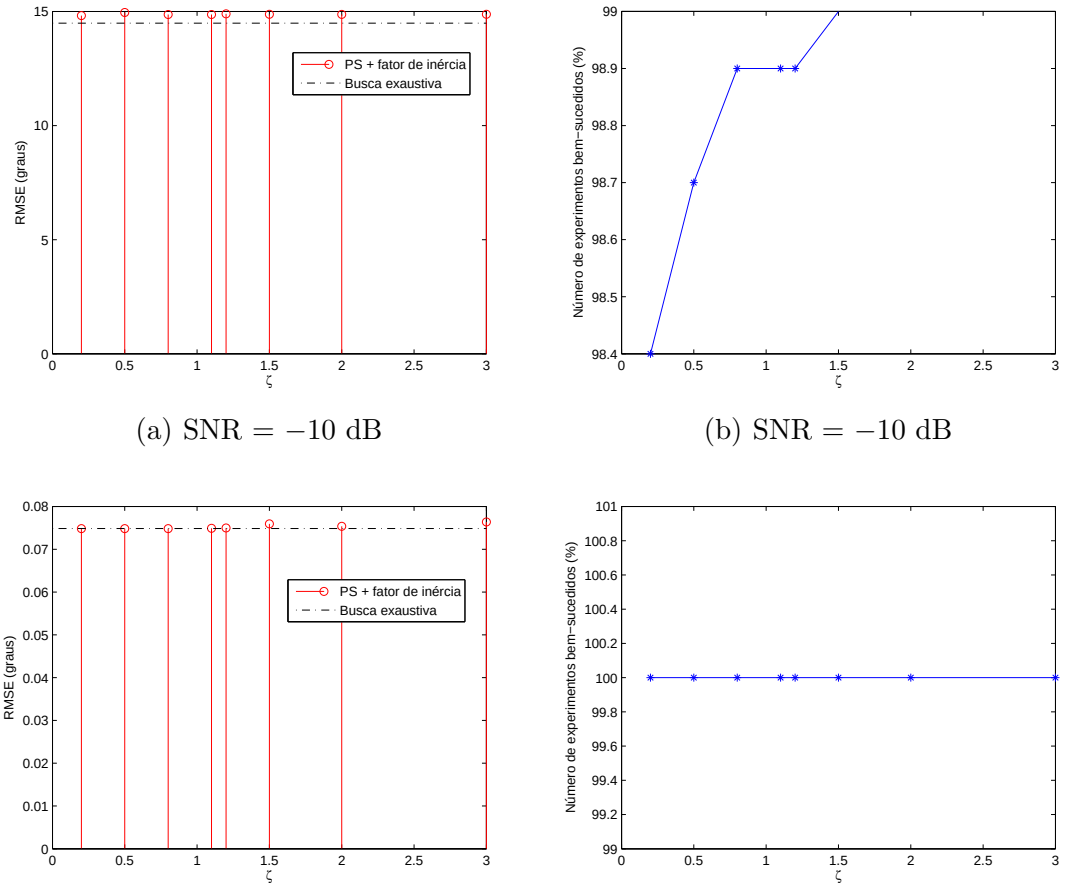
Observando a Expressão (3.27) referente à atualização do vetor velocidade de cada partícula, podemos apontar alguns aspectos relacionados ao impacto do parâmetro ζ no desempenho do algoritmo *particle swarm*:

1. se o valor de ζ for muito pequeno, isto significa que as parcelas responsáveis pela exploração local do espaço de busca predominam na atualização do vetor velocidade, de modo que, quando o algoritmo consegue localizar a região onde se encontra o ótimo global, rapidamente converge para este ponto. Todavia, há uma perda na capacidade de explorar novas regiões do espaço de busca, o que pode comprometer o desempenho do algoritmo em algumas situações, particularmente quando a superfície da função a ser otimizada possui um caráter multimodal acentuado.

2. por outro lado, se o valor de ζ for muito elevado, privilegia-se a parcela relacionada à busca global na atualização do vetor velocidade, o que tende a aumentar as chances de se encontrar a região onde o ótimo global está situado. Em contrapartida, isso pode prejudicar o refinamento local em torno dos pontos ótimos.

A Figura 4.15 apresenta as curvas de RMSE e do percentual de experimentos bem-sucedidos em função do parâmetro ζ obtidas como resultado dos testes de sensibilidade, considerando $N_P = 100$ e $\nu_{max} = 1{,}5$, para os casos em que a SNR é igual a -10 e 15 dB. No caso restante,

no qual a SNR é igual a -5 dB, o impacto do valor atribuído a ζ no desempenho do algoritmo foi praticamente desprezível.



(a) SNR = -10 dB (b) SNR = -10 dB

(c) SNR = 15 dB (d) SNR = 15 dB

Figura 4.15: Resultados dos testes de sensibilidade para o parâmetro ζ - *Particle Swarm* com fator de inércia.

Podemos observar que os aspectos destacados anteriormente com relação ao parâmetro ζ e sua influência no desempenho do algoritmo estão presentes nos resultados mostrados na Figura 4.15: 1) nota-se uma ligeira perda de qualidade na estimação dos ângulos de chegada quando o valor de ζ é elevado na Figura 4.15(c). 2) o algoritmo não consegue localizar o ótimo global em um número maior de experimentos quando o valor de ζ é pequeno, como mostra a Figura 4.15(b).

Destacamos que observações semelhantes a estas foram discutidas em (Shi e Eberhart, 1999). Neste trabalho, os autores também sugerem o uso de um fator de inércia cujo valor decresce linearmente ao longo das iterações. Desta forma, o algoritmo possui no início uma maior capacidade de explorar o espaço de busca e, próximo do final, quando a população possivelmente já localizou as principais regiões, sua capacidade de realizar um refinamento local aumenta, pois o valor de ζ é pequeno.

4.2.12 Particle Swarm com termo de constrição

Uma outra versão alternativa do *particle swarm*, proposta em 2002 (Clerc e Kennedy, 2002), inclui um termo de constrição χ na equação de atualização do vetor velocidade, como mostrado na Expressão (3.28). A introdução deste parâmetro visa ampliar as chances e a velocidade de convergência do algoritmo assim como controlar alguns comportamentos indesejados, como explosão. Além disso, elimina a necessidade de definir limites às componentes do vetor velocidade.

Embora o valor de χ, como discutido em (Clerc e Kennedy, 2002), esteja relacionado às constantes de aceleração L_1 e L_2, optamos por realizar testes de sensibilidade a fim de obter maiores informações acerca de sua influência no desempenho desta versão do algoritmo *particle swarm*. O número de partículas N_P também foi analisado por meio de testes de sensibilidade seguindo a metodologia descrita na Seção 4.2.1.

Os demais parâmetros, a saber, L_1, L_2 e $\max_{it}$, foram ajustados da mesma forma que na Seção 4.2.11. Ou seja, $L_1 = L_2 = 2{,}05$ e $\max_{it} = 500$. A seguir, apresentamos os resultados obtidos nos testes de sensibilidade realizados.

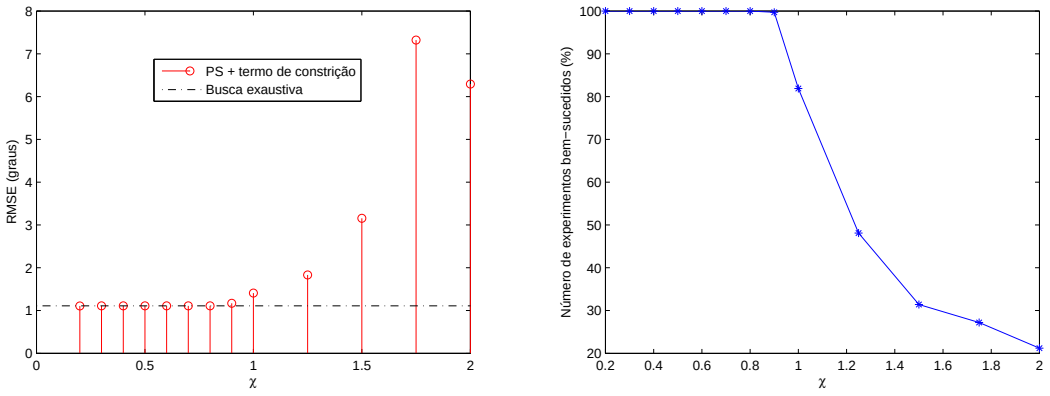
Parâmetro N_P - Número de indivíduos

Assim como nas versões anteriores do *particle swarm*, verificou-se através dos testes de sensibilidade, considerando $\chi = 0{,}729$, a mesma influência de N_P no desempenho do algoritmo:

à medida que o valor de N_P aumenta, nota-se que o erro de estimação, medido via RMSE, se aproxima da referência determinada pela busca em grade, enquanto a porcentagem de experimentos bem-sucedidos cresce.

Parâmetro χ - Termo de constrição

A Figura 4.16 exibe as curvas de RMSE e do percentual de experimentos bem-sucedidos em função do parâmetro χ obtidas como resultado dos testes de sensibilidade realizados de acordo com a metodologia exposta na Seção 4.2.1, considerando $N_P = 100$, para o caso em que a relação sinal-ruído é igual a -5 dB.



(a) SNR = -5 dB (b) SNR = -5 dB

Figura 4.16: Resultados dos testes de sensibilidade para o parâmetro χ - *Particle Swarm* com termo de constrição.

Podemos perceber na Figura 4.16 que, quando o valor de χ é relativamente baixo, especialmente na região em torno do valor 0,729, sugerido em (Clerc e Kennedy, 2002), o algoritmo *particle swarm* obteve um bom desempenho. Por outro lado, quando o valor de χ é alto, rapidamente ocorre uma degradação na qualidade das estimativas e também na capacidade de o algoritmo encontrar o ótimo global, haja vista a significativa redução na porcentagem de experimentos bem-sucedidos.

Ao observarmos a Expressão (3.28), podemos concluir que, quando o valor do parâmetro χ se eleva, crescem as chances de o vetor velocidade ter uma magnitude elevada, de modo que os

passos dados pelas partículas tornam-se cada vez maiores. Isto é indesejável, pois o algoritmo perde capacidade de exploração local. Além disso, as partículas, mesmo que encontrem uma região promissora do espaço de busca, serão forçadas a dar passos grandes, se afastando dos picos encontrados, inclusive do ótimo global. Por isso, sua capacidade de exploração do espaço de busca também fica comprometida nestas condições, o que explica a acentuada queda de desempenho observada na Figura 4.16.

4.2.13 Discussão

Os testes de sensibilidade realizados serviram para mostrar alguns indicativos da influência que os principais parâmetros dos algoritmos de computação natural possuem sobre o desempenho destas ferramentas no problema de estimação de direção de chegada. Em alguns casos, foi possível, inclusive, apontar intervalos ou até mesmo valores para o parâmetro sob análise que se mostram mais adequados.

Contudo, embora os testes de sensibilidade nos permitam analisar com profundidade a relação entre os principais parâmetros de cada algoritmo e o desempenho destas ferramentas, eles constituem apenas o passo inicial no estudo da aplicação destas ferramentas ao problema de estimação DOA.

Agora, tendo como base os resultados obtidos nos testes de sensibilidade, podemos atribuir valores aos parâmetros estudados e assim levantar, de forma similar à efetuada para os algoritmos MODE e MODEX na Seção 2.4.3, o desempenho de cada algoritmo de computação natural neste cenário DOA, expressando-os por meio de curvas de RMSE em função da relação sinal-ruído.

4.3 Análise de Desempenho

O passo seguinte no estudo da aplicação dos algoritmos de computação natural ao problema de estimação DOA consiste em determinar e expressar o desempenho de cada ferramenta consi-

derando o cenário clássico DOA apresentado nas Seções 2.3 e 4.2.1, para então realizarmos uma avaliação comparativa entre os algoritmos, de modo a evidenciar as potencialidades e limitações de cada método.

Para isto, é preciso primeiramente atribuir valores aos parâmetros dos algoritmos. Felizmente, uma vez que foram realizados testes de sensibilidade envolvendo os principais parâmetros de cada ferramenta, tal tarefa se tornou menos árdua. No entanto, para que seja feita uma comparação justa entre os algoritmos, é necessário garantir que os mesmos tenham condições iguais enquanto realizam a busca pelo ótimo global da função custo J_{ML}.

É justamente neste contexto que o número médio de avaliações da função de *fitness* (N_{fit}), cuja expressão foi deduzida para cada um dos algoritmos de computação natural no Capítulo 3, pode ser útil. Como mencionado na Seção 3.4.1, N_{fit} fornece uma indicação de quantos pontos do espaço de busca são amostrados e avaliados por um algoritmo até que o critério de parada seja atingido.

Assim sendo, ao definirmos *a priori* um valor para N_{fit}, imediatamente está estabelecido quantos pontos do espaço de busca o algoritmo poderá utilizar em sua busca pelo ótimo. Deste modo, se dois algoritmos tiverem seus parâmetros ajustados de tal forma que o valor de N_{fit} associado a cada um seja o mesmo, consideramos que estes algoritmos realizaram a busca em iguais condições, e, portanto, a comparação entre os respectivos desempenhos é válida.

Por isso, os parâmetros dos algoritmos de computação natural foram escolhidos de modo a assegurar aproximadamente o mesmo número de avaliações da função de *fitness*. Mas qual deve ser o valor de N_{fit}? Certamente não deve ser muito pequeno, pois assim o algoritmo pode não ter recursos suficientes para manipular sua população de soluções candidatas na tentativa de localizar o ótimo global. Por outro lado, também não pode ser muito elevado, já que almejamos reproduzir o desempenho da busca em grade com um custo computacional bem inferior, isto é, utilizando um número de pontos do espaço de busca bastante inferior àquele empregado pela busca em grade.

O procedimento adotado para a escolha do valor de N_{fit} e, conseqüentemente, para a

definição dos valores dos parâmetros de cada algoritmo, foi o seguinte: 1) a partir dos resultados dos testes de sensibilidade, extraímos um valor para cada parâmetro de cada algoritmo. 2) determinamos o valor correspondente de N_{fit} para cada algoritmo. 3) tendo em mente as observações mencionadas anteriormente quanto ao papel de N_{fit}, adotamos um valor adequado para N_{fit}. 4) introduzimos as devidas correções nos valores dos parâmetros, de modo a garantir o mesmo número médio de avaliações da função de *fitness* para todos os algoritmos.

Seguindo o procedimento descrito acima, escolhemos $N_{fit} = 100.000$. O Quadro 10 apresenta os valores atribuídos aos parâmetros de cada algoritmo de computação natural[2].

Quadro 10 Valores dos parâmetros

GA: $N_P = 150$, $p_c = 0{,}5$, $p_m = 0{,}05$, $N_t = 2$ e $\max_{it} = 1150$.
fitness sharing: $N_P = 150$, $p_c = 1{,}0$, $p_m = 0{,}02$, $\sigma_s = 10$ e $\max_{it} = 335$.
fitness scaling: $N_P = 150$, $p_c = 1{,}0$, $p_m = 0{,}02$, $\beta_s = 2$, $\sigma_s = 10$ e $\max_{it} = 335$.
clearing: $N_P = 150$, $p_c = 1{,}0$, $p_m = 0{,}03$, $\sigma_s = 10$, $\kappa = 1$ e $\max_{it} = 335$
$(\mu + \lambda)$-ES: $\mu = 150$, $\lambda = 20$ e $\max_{it} = 5000$.
(μ, λ)-ES: $\mu = 50$, $\lambda = 150$ e $\max_{it} = 700$.
DE: $N_P = 150$, $CR = 0{,}9$, $F = 0{,}5$ e $\max_{it} = 700$.
CLONALG: $N_P = 80$, $N_c = 5$, $\rho = 2$, $T_{Ab} = 20$, $p_{Ab} = 0{,}15$ e $\max_{it} = 250$.
opt-aiNet: $N_P = 40$, $N_c = 4$, $\rho = 2$, $\sigma_s = 1{,}4$, $T_{Ab} = 10$, $F_{dif} = 0{,}005$, $p_{Ab} = 0{,}2$ e $\max_{it} = 500$.
PS: $N_P = 150$, $\nu_{max} = 1{,}5$ e $\max_{it} = 700$.
PS + fator de inércia: $N_P = 150$, $\zeta = 1{,}1$, $\nu_{max} = 1{,}5$ e $\max_{it} = 700$.
PS + termo de constrição: $N_P = 150$, $\chi = 0{,}729$ e $\max_{it} = 700$.

É importante destacarmos que a rede *opt-aiNet* recebeu um tratamento um pouco distinto nesta etapa uma vez que não admite uma expressão fechada para o número médio de avaliações da função de *fitness* em função de seus parâmetros, pois, como já enfatizado na Seção 3.5.2, possui a capacidade de ajustar dinamicamente o tamanho da população. Neste caso, o controle do número de pontos utilizados durante a busca é feito no próprio critério de parada, de maneira que o algoritmo é interrompido quando N_{fit} avaliações forem atingidas.

Além disto, o procedimento empregado durante o ajuste dos parâmetros da *opt-aiNet* também foi diferente: agora, a métrica considerada é o número de avaliações da função de *fitness* realizadas em uma iteração do algoritmo. Assim, após definir valores para os parâmetros de acordo com os resultados dos testes de sensibilidade, determinamos o valor correspondente de

[2]Os parâmetros omitidos receberam os mesmos valores utilizados durante os testes de sensibilidade e podem ser encontrados na Seção 4.2.

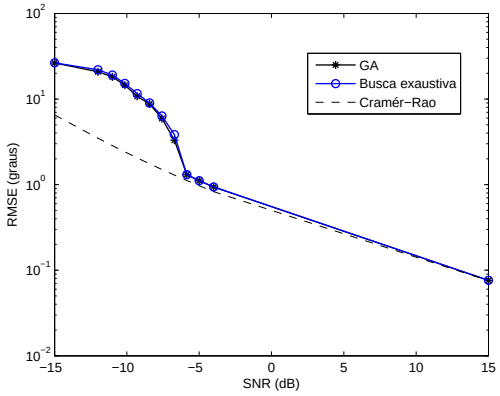
N_{fit} para uma iteração do algoritmo e, considerando alguns experimentos preliminares, projetamos o número de iterações que a *opt-aiNet* efetivamente utilizará até o critério de parada ser atingido. Se necessário, introduzimos algumas correções nos valores dos parâmetros para melhorar a relação entre a exploração do espaço de busca em uma iteração e o progresso do algoritmo ao longo das iterações.

As Figuras 4.17 e 4.18 apresentam as curvas de RMSE em função da relação sinal-ruído para cada um dos algoritmos de computação natural considerando os valores apresentados no Quadro 10 para os respectivos parâmetros. Novamente, para cada valor de SNR, foram realizados $N_e = 1000$ experimentos.

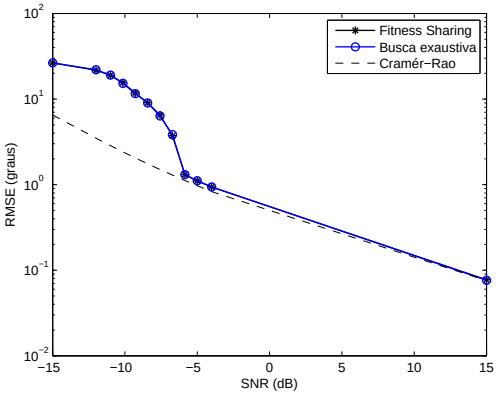
Em primeiro lugar, notamos na Figura 4.17(a) que o algoritmo genético atingiu um desempenho bastante próximo ao da busca em grade. Contudo, especialmente para dois valores de SNR, os valores RMSE referentes ao GA situam-se abaixo daqueles associados à busca em grade, o que indica que o GA falhou na identificação do ótimo global da função custo J_{ML} em alguns experimentos, fornecendo estimativas cujos valores situam-se mais próximos aos valores verdadeiros dos ângulos de chegada que o ótimo global. Durante os testes de sensibilidade, tal fato já havia sido verificado.

Podemos observar na Figura 4.17(e) que a estratégia evolutiva $(\mu + \lambda)$-ES também obteve um desempenho bastante próximo ao da busca em grade. Porém, assim como ocorreu com o algoritmo genético, foram obtidos valores RMSE abaixo da referência determinada pela busca em grade. Curiosamente, percebemos que estes dois algoritmos falham na estimação dos ângulos de chegada em alguns experimentos para o mesmo valor de SNR, o que sugere que as características da função custo J_{ML} nesta SNR são particularmente difíceis de serem trabalhadas por eles.

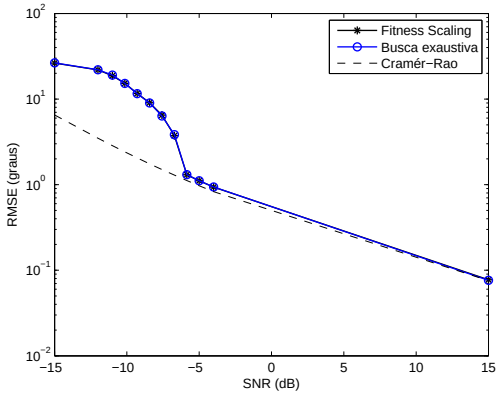
Notamos na Figura 4.17(f) que, embora tenha conseguido reproduzir o desempenho da busca em grade para valores elevados da SNR, o algoritmo (μ, λ)-ES não foi capaz de estimar corretamente os ângulos de chegada em diversos experimentos para valores de SNR inferiores a -5 dB, de modo que foram obtidos valores RMSE menores que os da busca em grade.



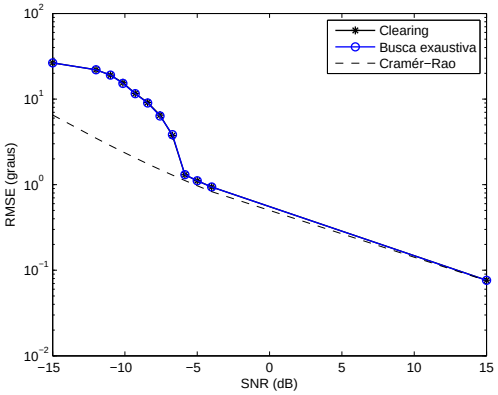
(a) Algoritmo Genético



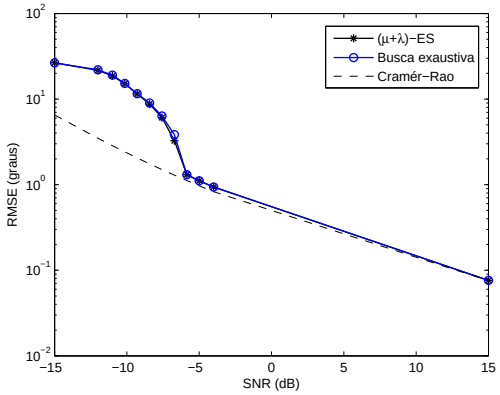
(b) *Fitness Sharing*



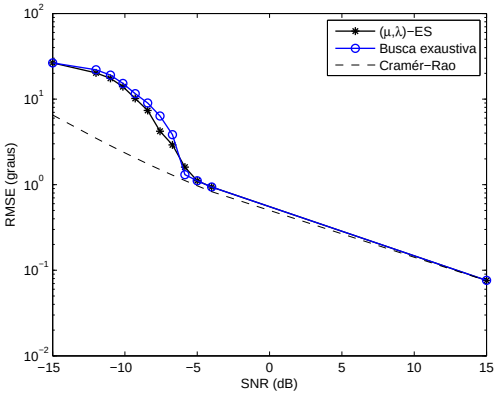
(c) *Fitness Scaling*



(d) *Clearing*

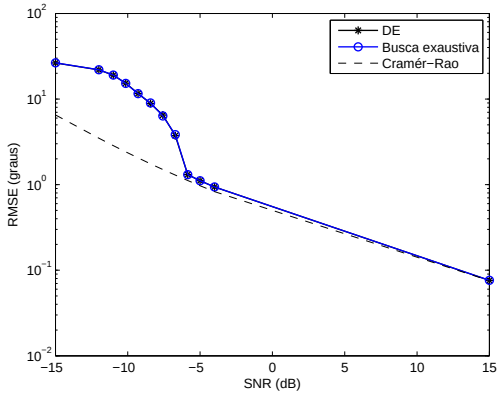


(e) $(\mu + \lambda)$-ES

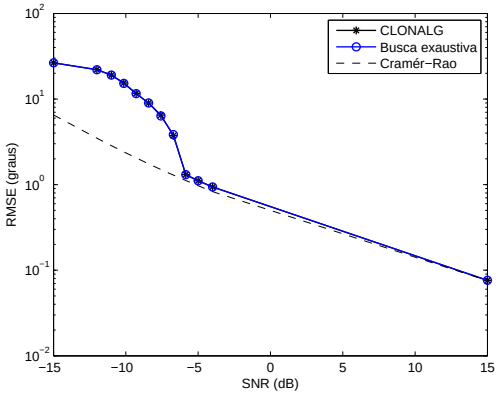


(f) (μ, λ)-ES

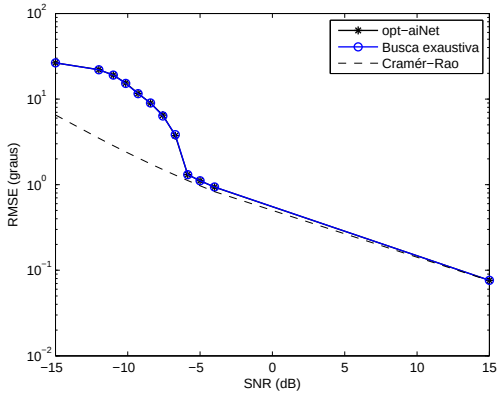
Figura 4.17: Curvas de RMSE em função da SNR.



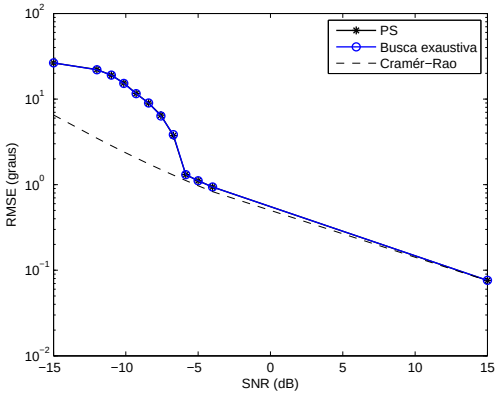
(a) Evolução Diferencial



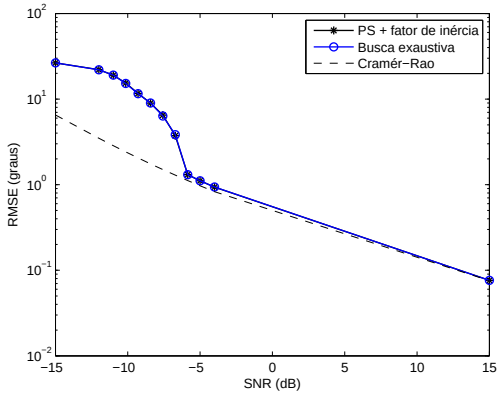
(b) CLONALG



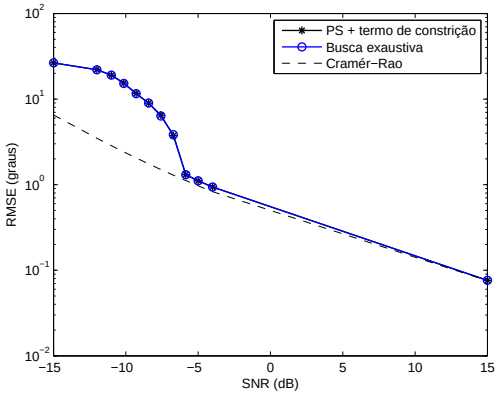
(c) *opt-aiNet*



(d) PS



(e) PS + fator de inércia



(f) PS + termo de constrição

Figura 4.18: Curvas de RMSE em função da SNR.

A razão deste comportamento está relacionada à dificuldade que o problema de estimação DOA, mediante o critério ML, impõe a qualquer ferramenta de otimização, a qual, a partir de uma única configuração de seus parâmetros, deve lidar com diferentes superfícies de otimização à medida que a SNR e a realização dos vetores de sinal e ruído variam.

Os demais algoritmos, a saber, *fitness sharing*, *fitness scaling*, *clearing*, DE, CLONALG, *opt-aiNet*, *particle swarm*, PS com fator de inércia e PS com termo de constrição, alcançaram um desempenho praticamente equivalente ao da busca em grade ao longo de toda a faixa de valores de SNR, o que significa que estas ferramentas conseguiram implementar o estimador ML para o problema de estimação DOA neste cenário.

Além disto, estas ferramentas superaram os métodos MODE e MODEX na região de limiar, pois foram capazes de acompanhar o desempenho da busca em grade, enquanto aqueles, como enfatizado na Seção 2.4.3, sofrem uma degradação de desempenho por causa do ruído, como mostrado na Figura 2.5.

Por fim, é fundamental destacarmos que as ferramentas mencionadas anteriormente que conseguiram acompanhar o desempenho da busca em grade alcançaram tal feito utilizando somente 100.000 pontos do espaço de busca, valor bastante inferior a $N_P = 3,2 \cdot 10^8$, que corresponde ao número de pontos considerados na busca em grade. Ou seja, os algoritmos de computação natural obtiveram êxito na estimação dos ângulos de chegada no cenário DOA considerado com um custo computacional, expresso pelo número de pontos utilizados durante a busca pelo ótimo global de J_{ML}, bem inferior ao da busca em grade.

4.4 Conclusão

Neste capítulo, os algoritmos de computação natural foram aplicados ao problema de estimação DOA considerando o cenário apresentado na Seção 2.3.

Primeiramente, foram realizados testes de sensibilidade, nos quais o desempenho de cada algoritmo, expresso em termos de RMSE e do percentual de experimentos bem-sucedidos, foi

monitorado enquanto o valor do parâmetro sob análise era alterado. Os resultados obtidos não apenas possibilitaram a visualização da influência que cada parâmetro exerce no desempenho de cada algoritmo, mas também guiaram a etapa de definição dos valores para os parâmetros.

Em seguida, sob condições iguais no que se refere ao número de pontos do espaço de busca amostrados e avaliados, determinou-se o desempenho de cada algoritmo neste cenário, tendo sido assim obtidas as curvas de RMSE em função da relação sinal-ruído apresentadas nas Figuras 4.17 e 4.18, as quais mostraram que os algoritmos *fitness sharing*, *fitness scaling*, *clearing*, DE, CLONALG, *opt-aiNet*, *particle swarm*, PS com fator de inércia e PS com termo de constrição determinam com precisão o mínimo global da função custo J_{ML} para uma ampla faixa de valores de relação sinal-ruído. Neste sentido, o desempenho destes algoritmos praticamente se igualou àquele oferecido pelo método de busca em grade. Vale ressaltar que todas estas técnicas amostraram e avaliaram um número de pontos do espaço de busca bastante inferior ao de uma metodologia de busca em grade, o que dá suporte à viabilidade da aplicação destas ferramentas.

No que tange à comparação com os algoritmos clássicos para a estimação DOA, os resultados também mostraram que estas ferramentas superam os métodos MODE e MODEX para um conjunto de valores de SNR, especialmente na região de limiar.

Em suma, este capítulo destinou-se à apresentação detalhada de todos os passos envolvidos na aplicação de algoritmos de computação natural ao problema de estimação DOA. No entanto, este estudo foi empreendido considerando um único cenário, de modo que diversos aspectos ainda precisam ser considerados para sustentar o uso destas ferramentas neste problema. Por exemplo, é preciso avaliar o impacto do aumento no número de fontes tanto no desempenho quanto no esforço computacional dos algoritmos de computação natural, particularmente no tocante ao número de pontos amostrados do espaço de busca. Esta e outras questões serão estudadas no próximo capítulo, cujo objetivo é apresentar indicativos de como estes algoritmos se comportam em outros cenários DOA, de modo a ressaltar a flexibilidade e o potencial destas técnicas na estimação dos ângulos de chegada.

Capítulo 5

Análise: Outros Cenários

5.1 Introdução

Os algoritmos de computação natural apresentados no Capítulo 3 foram aplicados ao problema de estimação DOA. O Capítulo 4 dedicou-se à análise criteriosa do desempenho destas ferramentas em um cenário consagrado na literatura. Os resultados obtidos mostraram que alguns destes algoritmos de fato são capazes de encontrar as estimativas ML para os ângulos de chegada, mesmo diante das dificuldades impostas pela presença de ruído.

No entanto, é evidente que o cenário tratado não é suficiente para credenciar os algoritmos de computação natural como alternativas viáveis aos métodos clássicos para a estimação DOA. Existem inúmeros outros aspectos que precisam ser considerados, dentre os quais destacamos:

1. Qual o impacto do aumento no número de fontes no desempenho dos algoritmos de computação natural? Além disto, será que o custo computacional destas ferramentas, à semelhança daquele referente a uma abordagem de busca em grade, também sofre um crescimento exponencial à medida que o número de fontes aumenta?

2. Se, porventura, outros valores forem atribuídos aos ângulos de chegada, os algoritmos serão capazes de manter a mesma excelência na estimação?

3. Qual seria o efeito da introdução de correlação entre as fontes de sinal sobre o desempenho destes algoritmos?

Por isso, compreendemos que é necessário empreender uma investigação acerca do desempenho dos algoritmos de computação natural em outros contextos DOA. Contudo, certamente não é possível exaurir todas as possibilidades. Por isso, selecionamos aquelas que nos parecem mais pertinentes, tendo como objetivo analisar como estes algoritmos se comportam em face a estes novos desafios e apresentar indicativos que sustentem a aplicação destas ferramentas ao problema de estimação DOA.

5.2 Outros Cenários

5.2.1 Ângulos de chegada distintos

A análise de desempenho para outros valores dos ângulos de chegada foi abordada considerando duas situações: primeiramente, estudamos o comportamento dos algoritmos quando o espaçamento angular entre as fontes de sinal é maior; em seguida, consideramos o mesmo espaçamento angular que o do cenário DOA utilizado no Capítulo 4, mas com valores para os ângulos de chegada situados na faixa próxima a 90°.

Primeiro Caso

Considere um cenário DOA equivalente ao apresentado na Seção 2.3, exceto pelos valores dos ângulos de chegada, agora definidos por $\boldsymbol{\phi} = [-10^\circ \; 45^\circ]^T$. Este novo cenário apresenta um espaçamento angular entre as fontes de sinal um pouco maior, o que, conforme já evidenciado na literatura, contribui para tornar um pouco mais simples a tarefa de estimar os ângulos de chegada.

Dentre as ferramentas de computação natural que se mostraram capazes de reproduzir o desempenho da busca em grade durante o estudo realizado no Capítulo 4, selecionamos três

algoritmos pertencentes a diferentes ramos de pesquisa dentro da Computação Natural para representarem este grupo durante a análise envolvendo este cenário: da computação evolutiva, escolhemos o algoritmo DE; dos sistemas imunológicos artificiais, o algoritmo CLONALG; e dos métodos baseados em inteligência de enxame, o algoritmo PS foi selecionado[1].

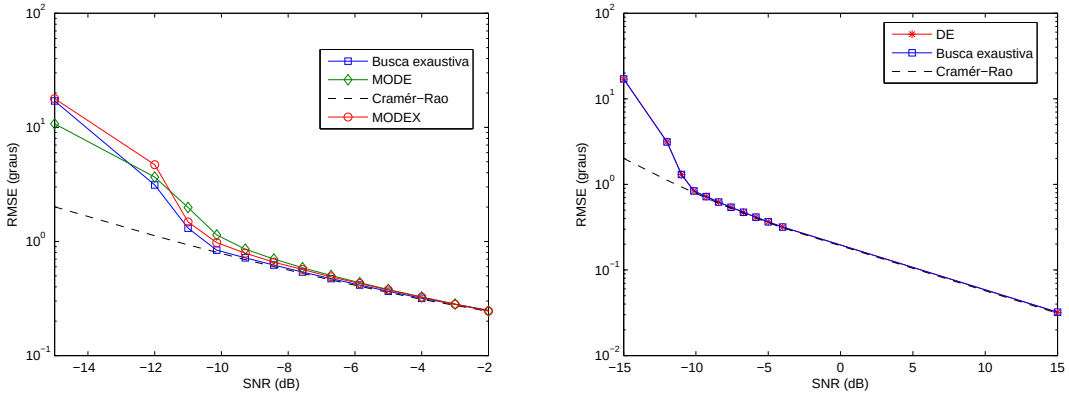
A princípio, uma vez que as condições do problema foram modificadas, deveríamos reajustar os parâmetros destes algoritmos a fim de torná-los melhor adaptados. No entanto, dado que o propósito deste estudo é extrair indicativos acerca do comportamento destas ferramentas em outras condições DOA e também avaliar o impacto que mudanças no cenário trazem ao desempenho dos algoritmos, optamos por utilizar os mesmos valores para os parâmetros especificados no Quadro 10, presente na Seção 4.3.

Como o número de fontes permaneceu inalterado, novamente podemos empregar uma busca em grade para estimar o ótimo global da função custo J_{ML} em todos os experimentos, a fim de estabelecer uma curva de RMSE em função da SNR que seja a referência de desempenho para qualquer método baseado no critério ML neste cenário. Novamente, foram utilizados $N_e = 1000$ experimentos para cada valor de SNR. Além disto, aplicamos também os algoritmos MODE e MODEX.

A Figura 5.1 apresenta as curvas de RMSE em função da relação sinal-ruído obtidas para os algoritmos de computação natural, assim como para a busca em grade e os métodos MODE e MODEX, além do limite de Cramér-Rao.

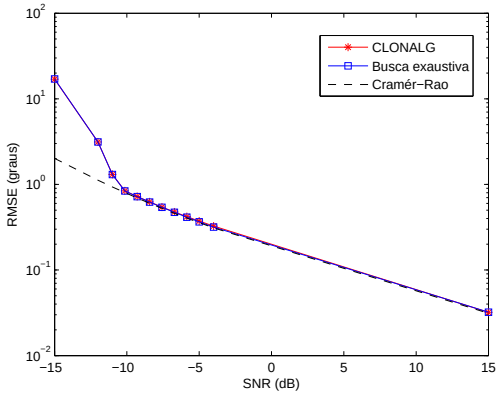
Em primeiro lugar, podemos notar na Figura 5.1(a) que os métodos MODE e MODEX, à semelhança do caso discutido na Seção 2.4.3, novamente sofrem uma degradação de desempenho à medida que a SNR diminui, se afastando do desempenho associado à busca em grade. No entanto, por causa do aumento do espaçamento angular entre as fontes, o grau desta degradação é menor se comparado com aquela observada na Figura 2.5 para o cenário DOA apresentado na Seção 2.3.

[1]Como nosso maior interesse neste capítulo é apontar indicativos do desempenho dos algoritmos de computação natural em outros cenários, optamos por fazer esta restrição.

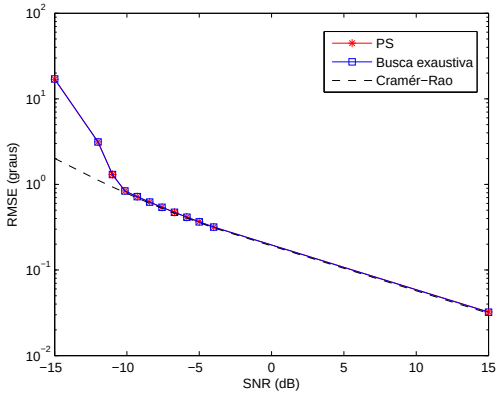


(a) MODE e MODEX

(b) DE



(c) CLONALG



(d) PS

Figura 5.1: Curvas de RMSE em função da relação sinal-ruído.

Quanto aos algoritmos de computação natural, observamos nas Figuras 5.1(b) a 5.1(d) que estas ferramentas novamente alcançaram os mesmos valores RMSE obtidos pela busca em grade. Portanto, foram capazes de localizar o ótimo global da função custo J_{ML} ao longo dos experimentos realizados, determinando as melhores estimativas segundo o critério ML.

Além disto, uma vez que os parâmetros destes algoritmos foram ajustados de acordo com os valores mostrados no Quadro 10, o número médio de avaliações da função de *fitness* realizadas por estas ferramentas novamente é igual a $N_{fit} = 100.000$. Logo, constatamos mais uma vez que os algoritmos DE, CLONALG e PS obtiveram um desempenho similar à busca em grade utilizando um número de pontos bastante inferior ao desta última abordagem, o que ressalta o potencial do uso destas técnicas no problema de estimação DOA.

Segundo Caso

Considere agora um cenário DOA similar ao apresentado na Seção 2.3, porém com $\boldsymbol{\phi} = [70^\circ\ 75^\circ]^T$. Nele, o espaçamento angular entre as fontes de sinal é preservado. Contudo, os ângulos de chegada estão situados numa região mais próxima a 90°.

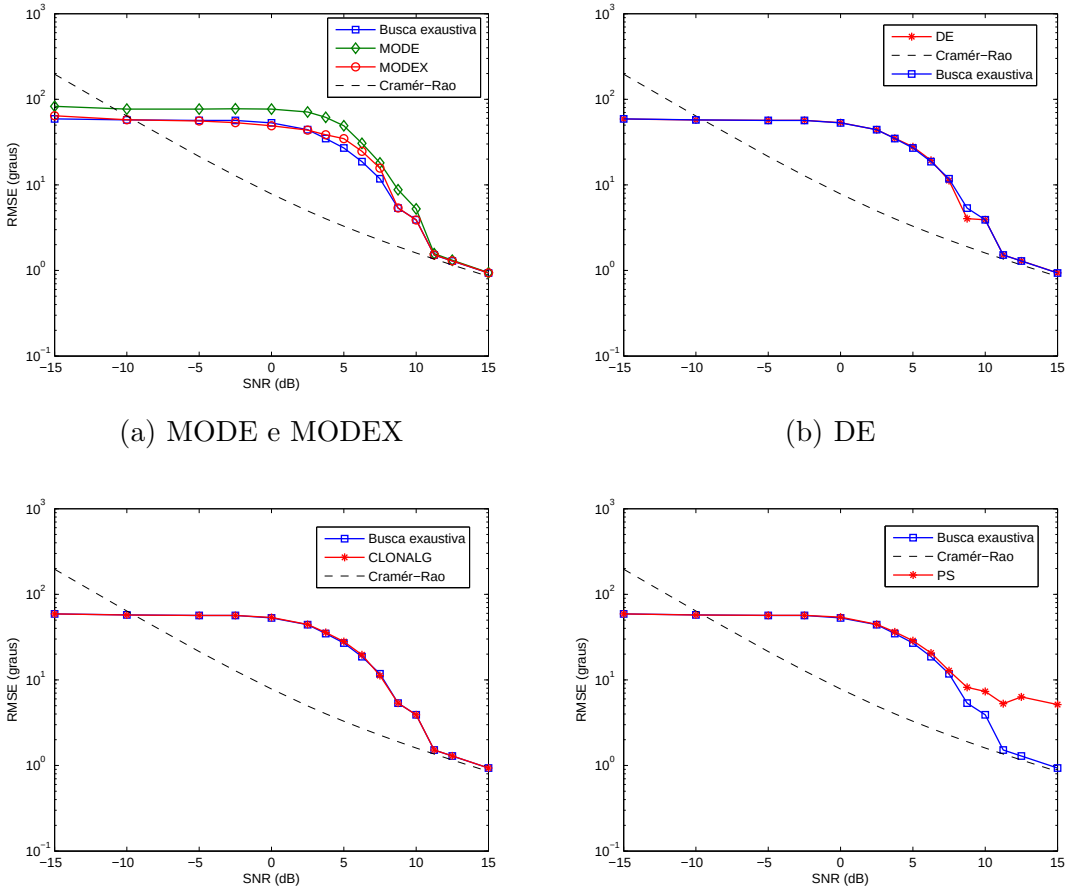
De forma intuitiva, podemos perceber o porquê de esta região impôr uma dificuldade maior à estimação DOA: quando nos aproximamos de 90°, os sinais atingem o arranjo quase que paralelamente ao seu eixo. Neste caso, pequenas variações nos valores dos ângulos de chegada não produzem variações apreciáveis nos valores das freqüências, afinal, a relação entre o ângulo e a freqüência envolve o seno do ângulo em relação à normal. Logo, a discriminação dos ângulos se torna mais imprecisa à medida que nos aproximamos de 90°, até que na condição paralela (90°), o arranjo não funciona.

A Figura 5.2 apresenta as curvas de RMSE em função da relação sinal-ruído obtidas para os algoritmos DE, CLONALG, e PS, assim como para a busca em grade e os métodos MODE e MODEX, além do limite de Cramér-Rao. Novamente, foram utilizados $N_e = 1000$ experimentos para cada valor de SNR. Além disto, os parâmetros dos algoritmos de computação natural, assim como feito no caso anterior, foram ajustados de acordo com o Quadro 10.

É possível observar na Figura 5.2(a) que o erro cometido pela busca em grade rapidamente se afasta do Limite de Cramér-Rao à medida que a SNR diminui. No entanto, quando a SNR assume valores inferiores a -10 dB, os valores RMSE obtidos por esta abordagem ficam abaixo do Limite de Cramér-Rao. Isto ocorre pelo seguinte motivo: em qualquer busca, seja em grade ou até mesmo um dos algoritmos de computação natural, os erros máximos estão limitados pela faixa $\pm 90^\circ$. Porém, a expressão do limite de Cramér-Rao não leva em conta esta limitação. Assim, o erro expresso pelo CRB pode crescer livremente.

Podemos notar também que os algoritmos MODE e MODEX novamente não conseguem atingir o mesmo desempenho oferecido por uma busca em grade. Por outro lado, a Figura 5.2(c) indica que o algoritmo CLONALG foi capaz de localizar as melhores estimativas segundo

o critério ML, obtendo um desempenho equivalente ao da busca em grade, com o diferencial de ter utilizado somente 100.000 pontos durante a busca. Neste sentido, sua aplicação ao problema DOA é novamente encorajada.



(a) MODE e MODEX (b) DE

(c) CLONALG (d) PS

Figura 5.2: Curvas de RMSE em função da relação sinal-ruído.

Outra ferramenta de computação natural que obteve um desempenho próximo ao da busca em grade foi o algoritmo DE, como é possível observar na Figura 5.2(b). No entanto, para um particular valor de SNR, este algoritmo obteve um valor RMSE inferior àquele oferecido pela busca em grade, o que, conforme já evidenciado e analisado no Capítulo 4, significa que, em alguns experimentos, o algoritmo falhou na identificação do ótimo global da função custo J_{ML}, fornecendo estimativas DOA que, embora estejam mais próximas dos valores verdadeiros dos ângulos de chegada que o próprio ótimo global, não atendem ao critério da máxima verossimi-

lhança.

Apesar disto, não devemos descartar a aplicação deste algoritmo ao problema DOA. Primeiramente, porque o algoritmo DE alcançou o mesmo desempenho que a busca em grade nos experimentos realizados durante o Capítulo 4 e também para o cenário tratado anteriormente. Em segundo lugar, o fato de o valor RMSE obtido pelo algoritmo DE não ser equivalente ao da busca em grade não significa que o mesmo falhou ao encontrar o ótimo global da função custo J_{ML} em todos os experimentos. Na realidade, basta que isto ocorra em alguns poucos experimentos, especialmente naqueles em que o ótimo global mais se afasta dos valores verdadeiros dos ângulos de chegada.

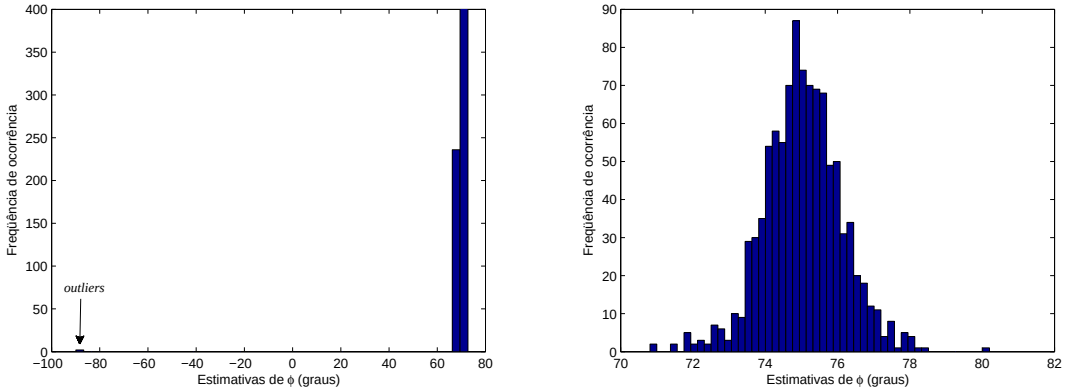
Além disto, é preciso considerar o papel dos parâmetros e do número de avaliações de *fitness* no comportamento do algoritmo. Os valores apresentados no Quadro 10, os quais também foram utilizados nos testes envolvendo este novo cenário, foram extraídos dos resultados obtidos durante os testes de sensibilidade relacionados ao cenário DOA descrito na Seção 2.3 e receberam correções a fim de garantir que o número total de avaliações de *fitness* realizadas por todos os algoritmos fosse igual a $N_{fit} = 100.000$. Contudo, não temos garantia alguma de que aqueles valores continuam sendo adequados neste novo cenário, tampouco de que o número de avaliações estipulado naquela ocasião é suficiente para este algoritmo resolver os problemas de otimização referentes a este novo cenário.

Por isso, salientamos que os resultados obtidos pelo algoritmo DE neste cenário, embora não tenham alcançado o mesmo nível de excelência que nos casos anteriores, ainda se mostram satisfatórios e capazes de fornecer um indicativo de que sua aplicação ao problema de estimação DOA continua sendo pertinente.

Por fim, podemos observar na Figura 5.2(d) que o algoritmo *particle swarm* não conseguiu atingir o mesmo nível de desempenho da busca em grade na região de valores de SNR superiores a 5 dB, dado que os valores RMSE obtidos por este algoritmo são maiores que aqueles associados à busca em grade. Em um primeiro momento, somos instigados a concluir que tal fato é uma clara evidência de que o PS não cumpriu a tarefa de encontrar o ótimo global da função custo

J_{ML}.

No entanto, como já enfatizado anteriormente, basta que em alguns poucos experimentos o algoritmo cometa um erro de estimação maior para comprometer a medida RMSE, já que a mesma se trata de uma média sobre todos os experimentos realizados. A fim de ratificarmos este argumento, apresentamos na Figura 5.3 os histogramas das estimativas obtidas pelo PS considerando os 1000 experimentos realizados para a SNR de 15 dB.



(a) Estimativas de $\phi = 70°$ (b) Estimativas de $\phi = 75°$

Figura 5.3: Histogramas das estimativas - *particle swarm*.

Podemos notar nas Figuras 5.3(a) e 5.3(b) uma predominância de estimativas centradas em torno dos valores verdadeiros dos ângulos de chegada, o que é um indício de que o algoritmo PS conseguiu encontrar o ótimo global da função custo J_{ML} nestes experimentos. Contudo, especialmente na Figura 5.3(a), observamos a presença de estimativas muito distantes do valor verdadeiro de ϕ, denominadas *outliers*. Estes *outliers* ocorreram somente em dois dos 1000 experimentos realizados, mas mesmo assim, são os responsáveis pela degradação observada na medida RMSE obtida para o algoritmo PS.

Através da análise dos histogramas fica evidente que o algoritmo PS, embora tenha obtido um desempenho inferior ao da busca em grade, na verdade conseguiu lidar com este novo cenário DOA, sendo capaz de encontrar o ótimo global da função custo na grande maioria dos experimentos. Os *outliers*, portanto, devem ser compreendidos como situações extremas,

que ocorrem em número muito reduzido, e que não refletem fielmente o real desempenho do algoritmo no problema de estimação DOA. Os motivos apontados anteriormente para justificar a queda de desempenho verificada para o algoritmo DE também são úteis para explicar o surgimento destes *outliers*.

Para complementarmos a análise relacionada ao *particle swarm*, apresentamos na Figura 5.4 a curva de RMSE em função da SNR para o algoritmo PS com termo de constrição utilizando os valores dos parâmetros definidos no Quadro 10. Como podemos verificar, o desempenho deste algoritmo praticamente equivale ao oferecido pela busca em grade. Isto indica que, considerando o mesmo conjunto de experimentos, esta versão do PS se mostra mais robusta que a versão original uma vez que, fazendo uso do termo de constrição χ, foi capaz de localizar o ótimo global da função custo J_{ML} naqueles experimentos em que os *outliers* haviam ocorrido.

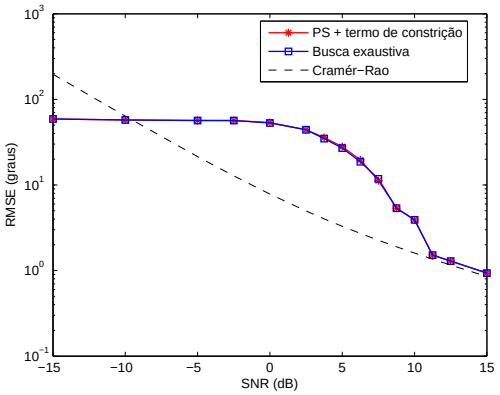


Figura 5.4: Curva de RMSE em função da relação sinal-ruído - PS com termo de constrição.

Discussão

Os dois cenários DOA tratados nesta Seção serviram para fornecer indicativos da influência que mudanças nos valores dos ângulos de chegada têm sobre o desempenho dos algoritmos de computação natural. Foram considerados dois casos: no primeiro, aumentamos o espaçamento angular entre as fontes; no segundo, deslocamos os ângulos para uma faixa de valores mais próxima de 90°.

Os algoritmos DE, CLONALG e PS foram selecionados como representantes dentre o grupo de ferramentas de computação natural que alcançaram o desempenho da busca em grade na análise realizada no Capítulo 4. Foi possível verificar que, no primeiro cenário, estes algoritmos novamente foram capazes de acompanhar o desempenho de uma busca em grade ainda que empregando durante a busca pelo ótimo um número de pontos bastante inferior ao utilizado por esta última abordagem.

O segundo cenário tratado serviu para ressaltar o potencial de aplicação do algoritmo CLONALG ao problema de estimação, uma vez que novamente, esta técnica conseguiu reproduzir o desempenho de uma abordagem de busca em grade. O algoritmo DE também alcançou um desempenho próximo ao da busca em grade, embora para um valor específico de SNR, o valor RMSE por ele obtido esteja situado abaixo da referência.

O algoritmo PS, por sua vez, embora tenha apresentado deficiências para alguns valores de SNR, também não deve ser descartado. Como argumentado, tais problemas não desqualificam estes dois últimos métodos uma vez que estão associados à ocorrência de *outliers* e também à própria escolha de parâmetros e do número máximo de avaliações da função de *fitness*. Além disso, ao contrapormos o desempenho deste algoritmo com aquele obtido pelo PS com o termo de constrição, notamos que este último algoritmo superou as dificuldades encontradas pelo PS original e foi capaz de acompanhar o desempenho da busca em grade.

Em suma, os resultados apresentados serviram tanto para ressaltar o potencial dos algoritmos de computação natural quanto para expôr alguns fatores que podem comprometer o desempenho destas ferramentas e que certamente precisam ser considerados durante sua aplicação real ao problema de estimação DOA.

5.2.2 Correlação entre fontes

Até o momento, os cenários considerados utilizaram a seguinte premissa: as fontes de sinal que incidem sobre o arranjo são independentes entre si. Nestes casos, os algoritmos de com-

putação natural mostraram ser alternativas viáveis a uma busca em grade, alcançando desempenhos muito próximos ao oferecido por esta última abordagem.

No entanto, sabe-se que a introdução de correlação entre as fontes traz dificuldades adicionais à estimação dos ângulos de chegada. Por exemplo, um método bastante conhecido, denominado MUSIC (em inglês, *Multiple Signal Classification*) (Schmidt, 1981) (Schmidt, 1986), ainda que seja estatisticamente consistente, isto é, capaz de oferecer estimativas cujos valores correspondem aos valores verdadeiros dos ângulos desde que haja um número suficiente de amostras ou a relação sinal-ruído seja suficientemente alta, sofre uma acentuada degradação de desempenho à medida que as fontes apresentam um coeficiente de correlação maior, até o ponto em que estas se tornam coerentes, situação na qual o MUSIC já não consegue realizar a estimação.

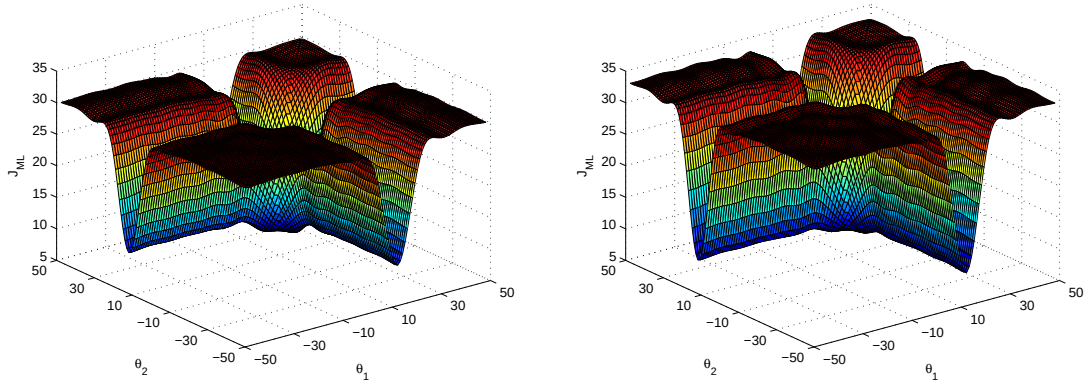
Por outro lado, foi mostrado que os algoritmos MODE e MODEX, na condição em que há correlação entre as fontes, embora apresentem uma suave degradação em relação ao caso descorrelacionado, continuam a realizar a estimação DOA de maneira suficientemente adequada (Stoica e Sharman, 1990) (A. Gershman e Stoica, 1999). Porém, como já fôra verificado na Seção 2.4.3, estes dois últimos métodos sofrem uma degradação em relação ao estimador ML à medida que a SNR diminui.

Logo, é pertinente verificarmos qual o impacto que a introdução de correlação entre as fontes de sinal tem sobre o desempenho dos algoritmos de computação natural. Para isto, considere o seguinte cenário DOA: $N = 10$, $M = 2$, $K = 100$, $\boldsymbol{\phi} = [10^\circ \; 15^\circ]^T$. Assim como adotado em (Stoica e Sharman, 1990), o coeficiente de correlação entre as fontes é igual a 0,98, de modo que a matriz de covariância de sinal é dada por

$$\mathbf{C} = \begin{bmatrix} 1 & 0{,}98 \\ 0{,}98 & 1 \end{bmatrix}.$$

A fim de ilustrarmos o efeito da correlação entre as fontes, apresentamos na Figura 5.5 as

superfícies da função custo J_{ML}, para a SNR de 0 dB, considerando primeiramente o caso em que as fontes são independentes e, posteriormente, quando temos a matriz de covariância $\mathbf{C}$ definida como acima.



(a) SNR = 0 dB, fontes independentes (b) SNR = 0 dB, fontes correlacionadas

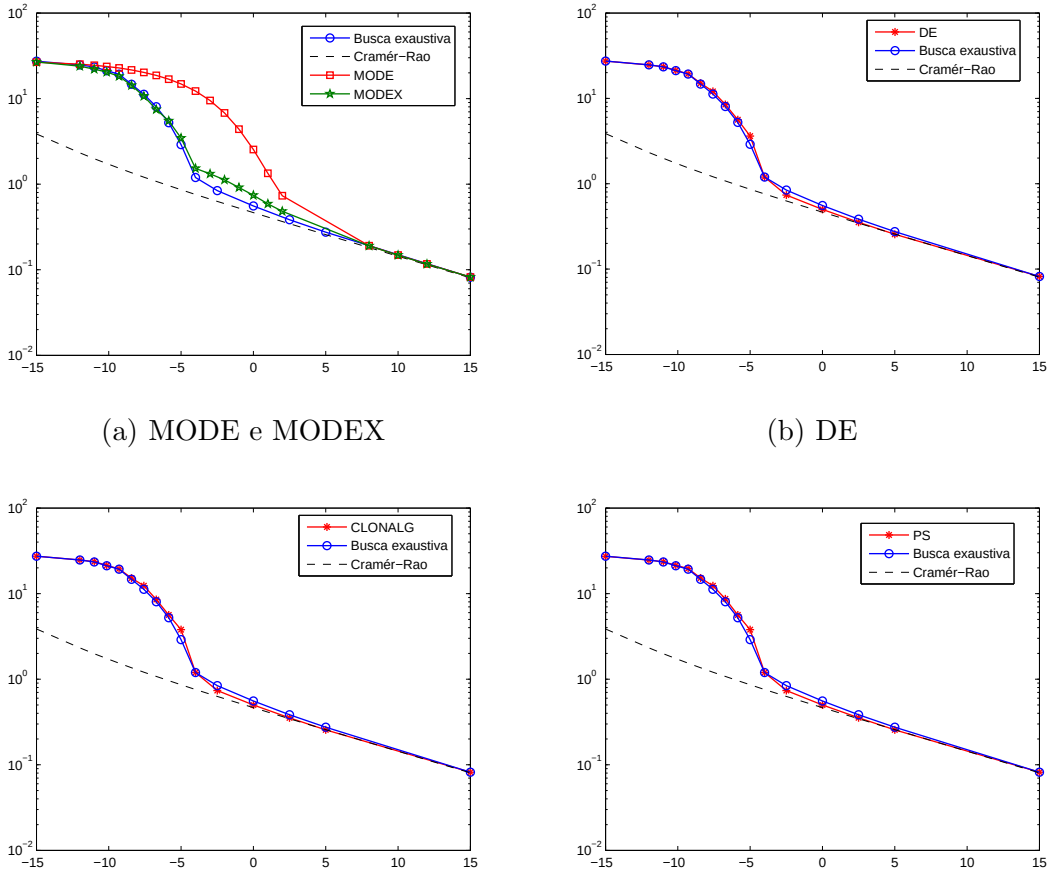
Figura 5.5: Superfícies da função ML Condicional para SNR = 0 dB, com e sem correlação.

Podemos notar na Figura 5.5 que a introdução de correlação entre as fontes não produz alterações significativas na superfície da função custo J_{ML}. Por causa disto, temos a expectativa de que, mais uma vez, os algoritmos de computação natural serão capazes de manter um desempenho bastante próximo ao do estimador ML ideal.

Assim como feito na Seção 5.2.1, os algoritmos DE, CLONALG e PS foram escolhidos como representantes das ferramentas de computação natural. A Figura 5.6 apresenta as curvas de RMSE em função da relação sinal-ruído obtidas para os algoritmos DE, CLONALG e PS, assim como para uma busca em grade e para os métodos MODE e MODEX, além do limite de Cramér-Rao. Para cada valor de SNR, foram utilizados $N_e = 1000$ experimentos. Quanto aos parâmetros dos algoritmos de computação natural, estes foram ajustados tendo como referência os valores estipulados no Quadro 10.

Os resultados mostrados na Figura 5.6 revelam que os algoritmos DE, CLONALG e PS alcançaram um desempenho em termos de RMSE praticamente equivalente ao da busca em grade ao longo da faixa de valores de SNR, superando também os métodos MODE e MODEX.

Podemos concluir, portanto, que o fato de as fontes de sinal apresentarem correlação entre si não compromete a estimação DOA realizada por estes algoritmos, o que fortalece a perspectiva de aplicação e uso destas ferramentas no problema de estimação DOA.



(a) MODE e MODEX

(b) DE

(c) CLONALG

(d) PS

Figura 5.6: Curvas de RMSE em função da relação sinal-ruído.

5.2.3 Número de fontes

Os cenários tratados nas seções anteriores possuem uma característica em comum: a presença de somente duas fontes de sinal. Isto possibilitou o uso de uma busca em grade para localizar, aproximadamente, a posição do ótimo global da função custo J_{ML} em todos os experimentos. Tendo em mãos a referência de desempenho, foi possível apontar quais algoritmos

de computação natural e em que condições foram capazes de implementar o estimador ML.

No entanto, se o problema de estimação DOA envolvesse unicamente cenários com apenas dois ângulos de chegada, todo o esforço empreendido durante as últimas décadas no desenvolvimento de métodos alternativos, como o MODE e o MODEX, e o próprio objetivo deste trabalho, perderiam um pouco de sua importância. Afinal, a busca em grade poderia ser empregada nestes casos, ainda que utilize um número elevado de pontos amostrados do espaço de busca.

Logo, é importante estudarmos a aplicação dos algoritmos de computação natural em cenários DOA onde a busca em grade já não pode ser empregada, a fim de verificar se nestas condições, estas ferramentas continuam a realizar adequadamente a estimação dos ângulos de chegada.

Todavia, ao partirmos para cenários que apresentam um número maior de fontes, nos deparamos com o seguinte dilema: ao mesmo tempo que almejamos ressaltar o potencial de aplicação destes algoritmos, através da confirmação de que, mesmo com um número maior de ângulos a serem estimados, tais métodos ainda implementam o estimador ML adequadamente, já não temos meios para assegurar indubitavelmente se cada ferramenta implementou ou não o estimador ML com perfeição, pois não há uma referência de desempenho a ser consultada.

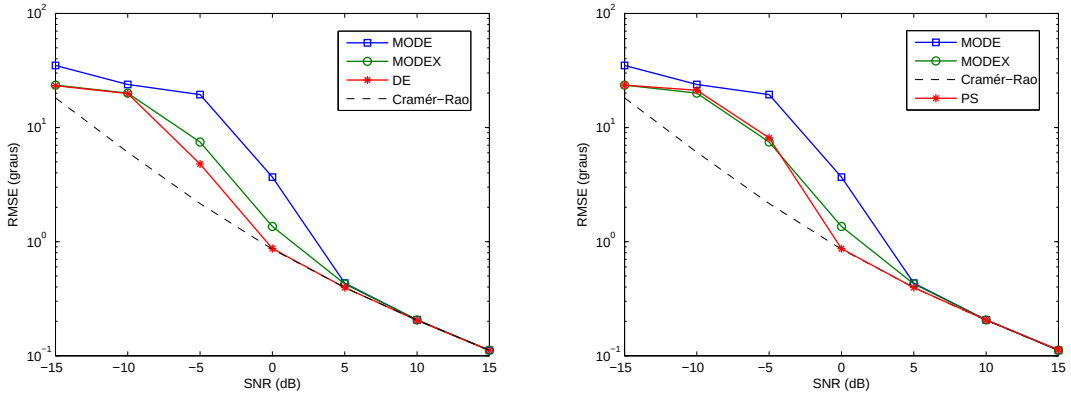
Ou seja, os cenários que poderiam ser bastante úteis para destacar o potencial de aplicação dos algoritmos de computação natural ao problema de estimação DOA são aqueles que pouco nos permitem afirmar acerca do real desempenho destas ferramentas e da qualidade de suas estimativas.

Apesar disto, ainda que sem uma referência universal de desempenho, realizamos um estudo do comportamento e desempenho dos algoritmos de computação natural quando o número de fontes aumenta. Primeiramente, consideramos um cenário no qual três fontes incidem no arranjo de sensores. Em seguida, a fim de apontar como o número de pontos amostrados do espaço de busca é influenciado pelo aumento no número de fontes, consideramos também um cenário com quatro fontes.

Três fontes

Considere o seguinte cenário DOA[2]: $N = 10$, $M = 3$, $K = 500$, $\mathbf{C} = \mathbf{I}$, $\boldsymbol{\phi} = [10^\circ\ 15^\circ\ 20^\circ]^T$.

Os algoritmos DE e PS foram escolhidos como representantes das ferramentas de computação natural nesta breve análise envolvendo o cenário descrito acima. A Figura 5.7 apresenta as curvas de RMSE em função da relação sinal-ruído obtidas para os algoritmos DE e PS, assim como para os métodos MODE e MODEX, além do limite de Cramér-Rao. Novamente, foram utilizados $N_e = 1000$ experimentos para cada valor de SNR. Além disto, os parâmetros dos algoritmos de computação natural foram ajustados segundo os valores apontados no Quadro 10.



(a) DE (b) PS

Figura 5.7: Curvas de RMSE em função da relação sinal-ruído.

Algumas observações interessantes podem ser extraídas a partir dos resultados mostrados na Figura 5.7. Primeiramente, podemos notar que, assintoticamente, o desempenho referente à cada algoritmo tende ao Limite de Cramér-Rao, e que, à medida que a SNR diminui, também sofre uma degradação e, por fim, se afasta do limite teórico. Nesta região, os valores RMSE obtidos pelos algoritmos de computação natural situam-se abaixo daqueles oferecidos pelos métodos MODE e MODEX, o que sugere que os primeiros superaram estes últimos na tarefa de estimar os ângulos de chegada.

[2]O número de *snapshots* K é maior, quando comparado ao utilizado nos cenários DOA anteriores, para garantir que o Limite de Cramér-Rao, na SNR de 15 dB, seja inferior a 0,01 graus.

Contudo, visto que não temos uma curva de RMSE que exprima o desempenho do estimador ML, não há como assegurar se de fato os algoritmos de computação natural conseguiram encontrar o ótimo global da função custo J_{ML} nos experimentos realizados. Esta dificuldade fica evidente quando focamos a SNR de -5 dB: nesta situação, não temos como garantir qual dos algoritmos de computação natural oferece o melhor desempenho, tampouco se algum deles efetivamente alcança o desempenho ML.

Ainda assim, os resultados mostrados na Figura 5.7 são úteis para ressaltar o potencial de aplicação de algoritmos de computação natural ao problema de estimação DOA, pois as curvas RMSE obtidas pelos algoritmos DE e PS apresentam características que se assemelham ao comportamento esperado para a curva RMSE associada ao estimador ML: no assintótico, equivalem ao Limite de Cramér-Rao; para valores muito pequenos de SNR, se aproximam da curva associada ao MODEX; na região de limiar, situam-se abaixo das curvas referentes ao MODE e ao MODEX.

Além disto, estes algoritmos utilizaram somente 100.000 pontos durante a busca pelo ótimo global de J_{ML}, valor muito inferior ao que seria necessário para uma busca em grade. Por isso, concluímos que, os resultados obtidos neste cenário dão respaldo à aplicação destas ferramentas ao problema de estimação DOA.

Quatro fontes

Finalmente, considere o cenário DOA definido a seguir: $N = 10$, $M = 4$, $K = 2000$, $\mathbf{C} = \mathbf{I}$, $\boldsymbol{\phi} = [10^\circ\ 15^\circ\ 20^\circ\ 30^\circ]^T$. Diferentemente do estudo realizado no cenário anterior, no qual verificamos o desempenho de alguns algoritmos de computação natural, agora temos por objetivo encontrar alguns indicativos de como se dá o crescimento do número de pontos utilizados durante a busca pelo ótimo global da função custo J_{ML} por um algoritmo de computação natural em função do número de fontes.

Para isto, restringimos as atenções ao algoritmo DE e a dois valores de SNR, a saber, 10 e 15 dB, para os quais o Limite de Cramér-Rao pode ser utilizado como a referência de desempenho.

A pergunta que desejamos responder é: Quantos pontos são necessários para que o desempenho do algoritmo DE, considerando os valores de 10 e 15 dB para a relação sinal-ruído e $N_e = 1000$ experimentos, seja equivalente ao Limite de Cramér-Rao?

Sabendo que o número de pontos utilizados pelo DE depende dos valores atribuídos a seus parâmetros, o ponto de partida para esta análise foi encontrar um ajuste para os parâmetros que seja suficiente para tornar o algoritmo DE capaz de encontrar o ótimo global da função custo. Felizmente, o Quadro 10 já nos fornece um ajuste, e a Tabela 5.1 apresenta os valores RMSE obtidos pelo algoritmo DE e os respectivos valores do Limite de Cramér-Rao.

	SNR	
	10 dB	15 dB
DE	0,2135	0,1125
Limite de Cramér-Rao	0,2066	0,1094

Tabela 5.1: Valores RMSE.

É possível perceber na Tabela 5.1 que os valores RMSE obtidos pelo algoritmo DE estão muito próximos àqueles referentes ao Limite de Cramér-Rao. Isto serve para reforçar o potencial de aplicação dos algoritmos de computação natural ao problema de estimação DOA: em um cenário no qual a busca em grade se torna absolutamente inviável, o algoritmo DE obteve um desempenho muito próximo ao desejado, mesmo empregando apenas 100.000 pontos durante a busca pelo ótimo global de J_{ML}.

Entretanto, é pertinente recordarmos que a análise realizada, tendo enfatizado somente o comportamento assintótico, não é suficiente para garantir que este conjunto de valores para os parâmetros também será adequado nos experimentos envolvendo outros valores de relação sinal-ruído. Contudo, certamente serve como indicativo de que os algoritmos de computação natural, ao contrário do que acontece com a busca em grade, não requerem um número de pontos para a busca que cresce exponencialmente com um aumento no número de fontes, o que viabiliza sua aplicação.

5.3 Conclusão

Este capítulo destinou-se a uma investigação relativa ao desempenho dos algoritmos de computação natural quando aplicados ao problema de estimação de direção de chegada em outras condições além daquelas contempladas no Capítulo 4.

Primeiramente, avaliamos se os valores atribuídos aos ângulos de chegada exercem significativa influência no desempenho destas ferramentas. Foi possível perceber que, no caso em que o espaçamento entre os ângulos é maior, os algoritmos de computação natural, assim como ocorreu no cenário DOA tratado no Capítulo 4, reproduziram o desempenho do estimador ML ideal. Porém, no caso em que os ângulos foram alocados numa região mais próxima a 90°, constatamos que estas ferramentas depararam-se com dificuldades adicionais que, em algumas raras situações, os conduziram a estimativas muito distantes dos valores verdadeiros dos ângulos de chegada.

Não obstante, os resultados obtidos não apenas serviram para ressaltar o potencial de aplicação destas ferramentas como também para apontar algumas preocupações que se mostram necessárias ao correto estudo do desempenho destes algoritmos.

Em seguida, passamos à análise do impacto que a introdução de correlação entre as fontes tem sobre o comportamento e desempenho dos algoritmos de computação natural. Os resultados obtidos confirmaram a expectativa preliminar de que a presença de fontes correlacionadas não influenciaria de forma significativa estas ferramentas. Com efeito, verificamos que, nesta condição, os algoritmos de computação natural conseguem realizar adequadamente a estimação dos ângulos de chegada, sendo novamente capazes de acompanhar o desempenho de uma busca em grade.

Por fim, estudamos como os algoritmos de computação natural se comportam quando enfrentam cenários com um número maior de fontes. A análise feita, embora desprovida de uma referência para o desempenho do estimador ML, foi útil para destacar a flexibilidade que estas ferramentas possuem, pois empregando o mesmo número de pontos durante a busca pelo ótimo

que aquele referente aos cenários com duas fontes, alcançaram um desempenho que atende aos requisitos esperados do estimador ML, ainda que não possamos garantir que este último tenha sido atingido efetivamente.

No próximo capítulo, propomos uma forma de melhorar o desempenho dos algoritmos de computação natural, tornando-os mais eficientes através do uso de um filtro apropriado capaz de reduzir a componente de ruído na matriz de covariância do sinal recebido.

Capítulo 6

Uso de Filtragem

6.1 Introdução

Os experimentos realizados ao longo dos Capítulos 4 e 5 proporcionaram uma visão abrangente acerca das limitações e potencialidades dos algoritmos de computação natural na estimação DOA considerando diferentes cenários. Os resultados obtidos forneceram indicativos de que algumas destas ferramentas alcançaram o desempenho do estimador ML.

No entanto, é importante recordarmos que, até o momento, os algoritmos estudados não exploram nenhuma informação adicional sobre o problema de estimação DOA, senão a própria função custo J_{ML} associada ao critério ML.

O fato de não utilizarem outras informações específicas deste problema não impediu que alguns destes algoritmos realizassem adequadamente a estimação dos ângulos de chegada. Contudo, à medida que um algoritmo se especializa em resolver um determinado problema, por meio da introdução de operadores ou mecanismos de busca dedicados, o seu desempenho tende a ser aprimorado.

Isto nos remete ao espírito dos algoritmos meméticos, os quais se caracterizam pela sinergia entre métodos de busca global e heurísticas de caráter local na solução de problemas (Moscato, 1989). Por isso, é bastante pertinente avaliarmos a possibilidade de incorporar aos algoritmos

de computação natural as informações que dispomos a respeito do problema de estimação DOA, na tentativa de torná-los mais habilitados e até mesmo mais eficientes.

Neste capítulo, apresentamos sucintamente como uma proposta de filtragem em estimação DOA pode ser utilizada em conjunto com os algoritmos de computação natural de modo a auxiliá-los na tarefa de encontrar as melhores estimativas DOA segundo o critério ML.

6.2 Processo de Filtragem

Seja $\mathbf{y}_k \in \mathcal{C}^{N\times 1}$ a seqüência contendo as amostras dos sinais recebidos pelo arranjo no k - ésimo *snapshot*. Considere também um filtro linear de resposta ao impulso finita (em inglês, *finite impulse response* (FIR)) cuja função de sistema é definida por

$$H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + \cdots + h_L z^{-L}, \tag{6.1}$$

onde $L \leq (N-1)$ corresponde à ordem do filtro.

Ao submetermos a seqüência $\mathbf{y}_k$ ao filtro, obtemos na saída uma seqüência $\mathbf{z}_k$ de comprimento $N+L$ dada pela convolução entre $\mathbf{y}_k$ e a resposta ao impulso do filtro FIR, como mostra a expressão abaixo:

$$\mathbf{z}_k = \mathbf{H}\mathbf{y}_k\,, \tag{6.2}$$

onde $\mathbf{z}_k \in \mathcal{C}^{(N+L)\times 1}$ é o vetor das saídas do filtro (*snapshot* filtrado) e $\mathbf{H} \in \mathcal{C}^{(N+L)\times N}$, de posto cheio igual N, é a matriz de convolução

$$\mathbf{H} = \begin{bmatrix} h_0 & 0 & \cdots & 0 \\ h_1 & h_0 & \cdots & 0 \\ \vdots & \vdots & & \\ h_L & h_{L-1} & \cdots & \\ 0 & h_L & \cdots & \\ \vdots & \vdots & & \\ 0 & 0 & \cdots & h_L \end{bmatrix}. \tag{6.3}$$

Uma vez que o problema de estimação DOA, como apresentado no Capítulo 2, está formulado como um problema de estimação de freqüência, é possível reduzir a componente de ruído presente na matriz de covariância $\hat{\mathbf{R}}_{\mathbf{y}}$ por meio de um filtro FIR multibanda cuja resposta em freqüência $H_f(\omega)$ deve satisfazer as seguintes restrições:

1. $|H_f(\omega)|$ deve ser igual a 1 nas freqüências ω_m, $m = 1, \ldots, M$.

2. $|H_f(\omega)|$ deve ser inferior a 1 nas freqüências $\omega \neq \omega_m$, $m = 1, \ldots, M$.

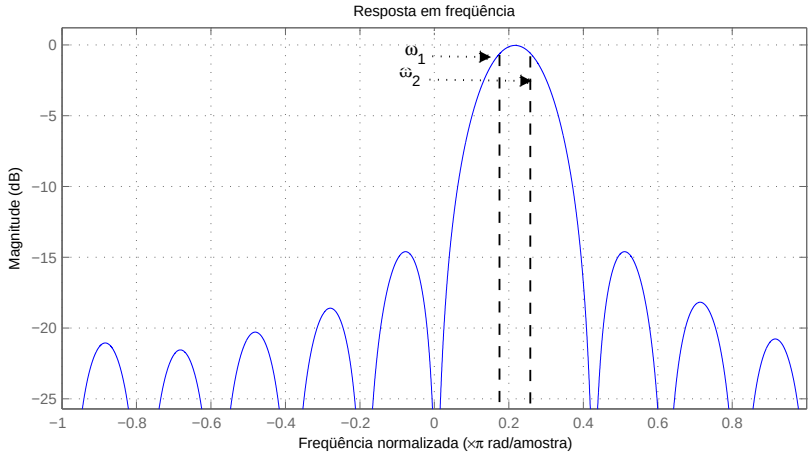
Desejamos encontrar a ordem L e os coeficientes h_i do filtro FIR que maximizam a relação sinal-ruído média na saída do filtro, ou seja, que produzam a maior atenuação da potência do ruído presente no *snapshot* filtrado $\mathbf{z}_k$. Em (Krummenauer, 2007) (Krummenauer, Cazarotto, Lopes, Larzabal, e Forster, 2010), foi demonstrado que, usando $L = (N - 1)$, os coeficientes ótimos do filtro são dados pelo autovetor da matriz $\hat{\mathbf{R}}_{\mathbf{y}}$ associado ao seu maior autovalor.

A Figura 6.1 apresenta o módulo das respostas em freqüência do filtro FIR otimizado segundo o critério da maximização da relação sinal-ruído média, considerando o cenário DOA apresentado na Seção 2.3, para as SNR de 15 dB, 0 dB e -10 dB, respectivamente. Os ângulos de chegada definidos por $\boldsymbol{\phi} = [10^\circ \; 15^\circ]^T$ geram as freqüências normalizadas em relação a π, $\omega_1 = 0{,}1736$ e $\omega_2 = 0{,}2588$, respectivamente.

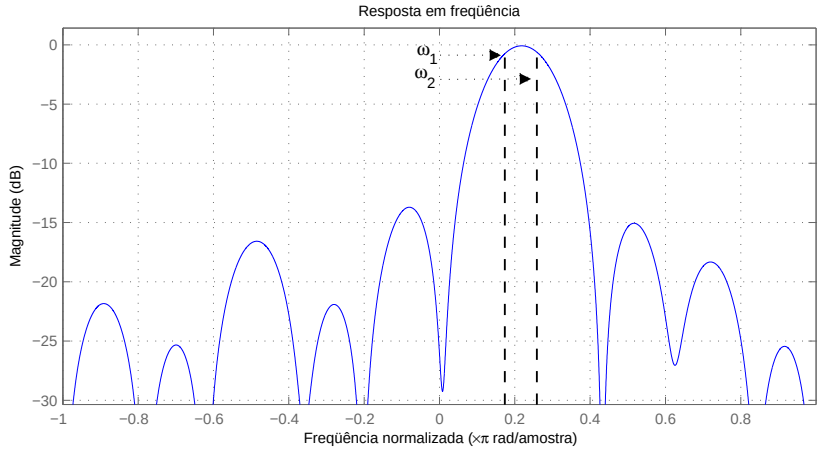
Como pode ser observado nas Figuras 6.1(a), 6.1(b) e 6.1(c), o filtro FIR preserva as características desejadas para a resposta em freqüência mesmo para valores baixos de relação sinal-ruído.

6.3 Proposta de uso da filtragem

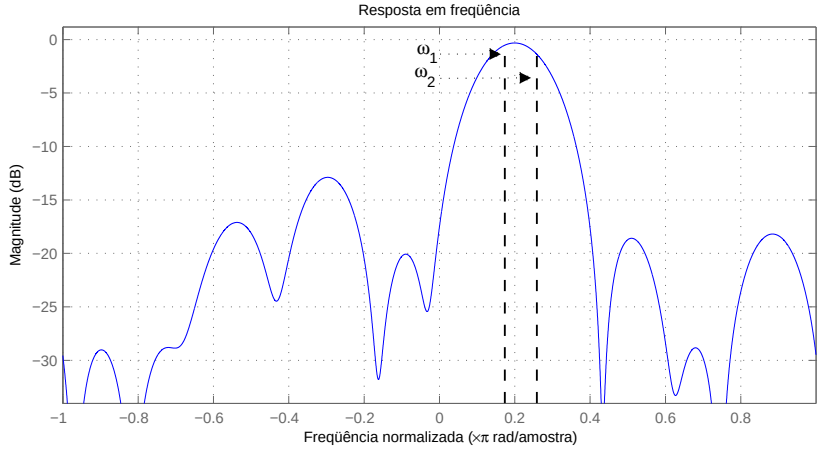
O processo de filtragem descrito na Seção 6.2 correlaciona o ruído em sua saída e transforma a parcela de sinal dos dados recebidos. Por conta disto, em (Krummenauer, 2007) (Krummenauer et al., 2010), uma nova expressão para o estimador ML adaptada a estas



(a) SNR = 15 dB



(b) SNR = 0 dB



(c) SNR = −10 dB

Figura 6.1: Resposta em freqüência do filtro FIR otimizado para relações sinal-ruído de 15 dB, 0 dB e −10 dB.

condições é derivada, substituindo aquela definida pela Expressão (2.37) no processo de seleção das raízes feito pelo algoritmo MODEX.

Neste trabalho, este processo de filtragem foi empregado de uma forma diferente: em vez de submetermos os dados coletados ao filtro FIR, de modo a obter uma nova expressão para o estimador ML, nosso interesse concentra-se apenas na informação presente na resposta em freqüência do filtro. Podemos observar na Figura 6.1 que as freqüências verdadeiras, isto é, aquelas correspondentes aos verdadeiros ângulos de chegada, situam-se na região de maior ganho do filtro.

Com base nesta evidência, nossa proposta é utilizar a resposta em freqüência do filtro FIR otimizado como função densidade de probabilidade a ser empregada no processo de amostragem do espaço de busca durante a composição da população inicial de soluções candidatas dos algoritmos de computação natural. Assim, em vez de extraírmos as amostras segundo uma distribuição uniforme, o que significa que todos os ângulos no intervalo $(-90°, 90°)$ são igualmente prováveis, agora privilegiamos os ângulos relacionados às freqüências presentes nas faixas de maior ganho da resposta em freqüência do filtro.

A Figura 6.2 exibe o histograma das estimativas presentes em uma população contendo um milhão de soluções candidatas amostradas segundo a resposta em freqüência do filtro FIR para a SNR de 15 dB, considerando o cenário DOA apresentado na Seção 2.3. Podemos notar uma grande semelhança entre o histograma das estimativas e a resposta em freqüência do filtro FIR mostrada na Figura 6.1(a). Por este motivo, as estimativas concentraram-se em torno dos valores verdadeiros dos ângulos de chegada.

A fim de ilustrarmos o efeito de se utilizar a resposta em freqüência do filtro FIR otimizado como função densidade de probabilidade, mostramos na Figura 6.3 a distribuição dos indivíduos que compõem a população inicial sobre a superfície de nível da função custo J_{ML} para a SNR de 15 dB. No primeiro caso, a amostragem foi feita segundo uma distribuição uniforme, enquanto no segundo, a resposta em freqüência do filtro FIR otimizado foi empregada.

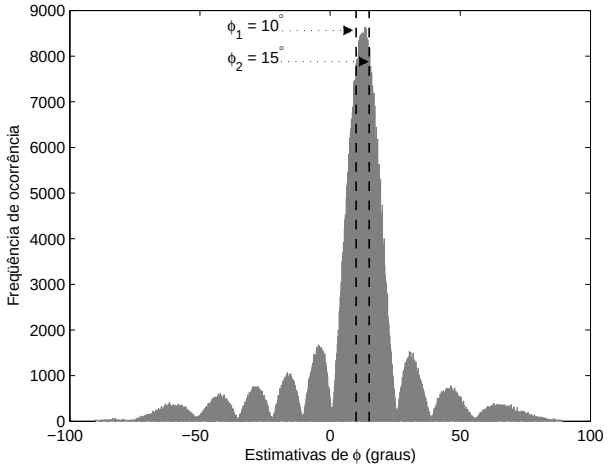
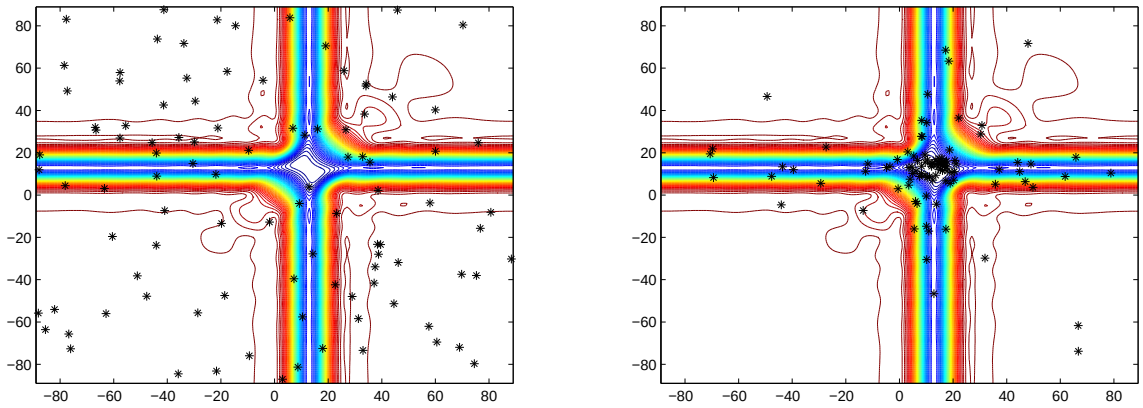


Figura 6.2: Histograma das estimativas presentes na população inicial amostrada segundo a resposta em freqüência do filtro FIR para SNR = 15 dB.



(a) Inicialização uniforme

(b) Inicialização com filtro FIR

Figura 6.3: População inicial amostrada segundo uma distribuição uniforme e segundo a resposta em freqüência do filtro FIR para SNR = 15 dB.

É possível observar na Figura 6.3 que os indivíduos da população amostrada segundo a resposta em freqüência do filtro FIR se concentram na região associada aos valores verdadeiros dos ângulos de chegada. Neste caso, podemos afirmar que os algoritmos de computação natural tenderiam a localizar o ótimo global com mais facilidade. Logo, temos um indicativo de que o uso da filtragem pode ser bastante interessante.

Porém, será que o filtro FIR sempre favorece o desempenho dos algoritmos? Como discutido

na Seção 2.3, à medida que a SNR diminui, o ótimo global pode se afastar cada vez mais dos valores verdadeiros dos ângulos de chegada. Nestas situações, como mostrado na Figura 6.1(c), a resposta do filtro FIR continua privilegiando a região associada aos valores verdadeiros dos ângulos de chegada. Contudo, ao mesmo tempo, pode inibir a amostragem de pontos na região onde o ótimo global da função custo J_{ML} está situado, visto que o ganho nesta faixa de freqüências é bem menor, o que, conseqüentemente, pode prejudicar o desempenho dos algoritmos, já que a população inicial tenderá a se concentrar em regiões distantes do ponto ótimo.

Ou seja, intuitivamente, é de se esperar que o ganho obtido com o uso da resposta em freqüência do filtro FIR otimizado tenda a cair à medida que a SNR é reduzida. No entanto, desejamos que, mesmo nestas situações, isto não comprometa o desempenho dos algoritmos de computação natural.

Na próxima seção, avaliamos o impacto da proposta de uso da filtragem sobre o desempenho dos algoritmos de computação natural.

6.4 Resultados

Considere o cenário DOA descrito na Seção 2.3: $N = 10$, $M = 2$, $K = 100$, $\boldsymbol{\phi} = [10^\circ\ 15^\circ]^T$ e $\mathbf{C} = \mathbf{I}$. No Capítulo 4, foi verificado que, neste cenário, diversos algoritmos de computação natural conseguem atingir o mesmo nível de desempenho de uma busca em grade, sendo, portanto, capazes de encontrar as melhores estimativas segundo o critério ML.

Apesar disto, é bastante pertinente ponderarmos as seguintes questões: Caso a população inicial fosse gerada tendo a resposta em freqüência do filtro FIR otimizado como função densidade de probabilidade, qual seria o desempenho destes algoritmos? Afinal, haveria algum ganho ao utilizar esta abordagem?

Com o propósito de esclarecermos estas questões, primeiramente, selecionamos os algoritmos DE e CLONALG como representantes dos algoritmos de computação natural que, neste

cenário, alcançaram o desempenho do estimador ML. Em seguida, considerando tanto o caso de inicialização uniforme, quanto o caso em que a resposta em freqüência do filtro FIR otimizado é empregada, aplicamos estes dois algoritmos ao problema de estimação DOA e os respectivos desempenhos foram monitorados à medida que o valor da SNR era reduzido. Contudo, ao invés de os expressarmos através da medida RMSE, as seguintes métricas foram utilizadas:

1. Percentual de experimentos bem-sucedidos: através desta métrica, é possível observar se a proposta de uso de filtragem afeta a capacidade de o algoritmo encontrar o ótimo global.

2. Número de avaliações da função de *fitness* até que o ótimo global seja localizado: nos experimentos em que ambas versões dos algoritmos DE e CLONALG de fato encontram o ótimo global, esta métrica é útil para apontar o custo requerido por cada uma delas até completar a tarefa.

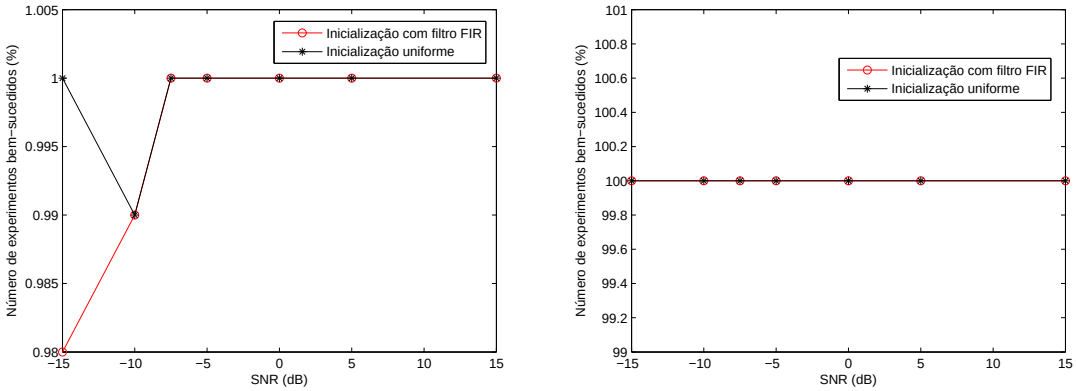
3. Ganho percentual no número de iterações necessárias para a correta localização do ótimo global: $G_{it} = 100(1 - \frac{n_1}{n_2})$, onde n_1 e n_2 definem o número de iterações requeridas pelo algoritmo nas versões com inicialização via filtro FIR e distribuição uniforme, respectivamente.

As Figuras 6.4 a 6.6 apresentam todas as curvas obtidas para os algoritmos DE e CLONALG. Para cada valor de SNR, foram realizados $N_e = 100$ experimentos. Os valores atribuídos aos parâmetros de cada algoritmo estão mostrados no Quadro 11.

Quadro 11 Valores dos parâmetros

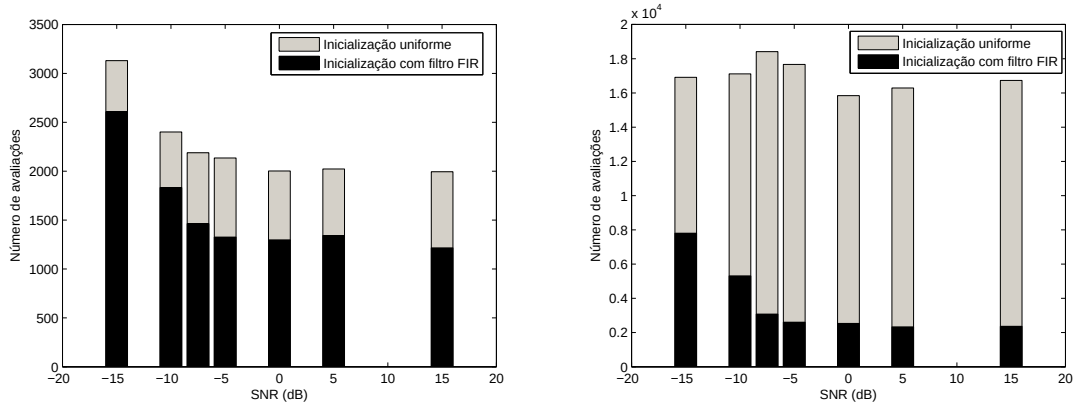
DE: $N_P = 100$, $CR = 0{,}9$, $F = 0{,}5$ e $\max_{it} = 200$.
CLONALG: $N_P = 80$, $N_c = 5$, $\rho = 2$, $T_{Ab} = 20$, $p_{Ab} = 0{,}15$ e $\max_{it} = 250$.

Em primeiro lugar, podemos notar na Figura 6.4 que o uso da resposta em freqüência praticamente não afetou os percentuais de experimentos bem-sucedidos associados aos algoritmos DE e CLONALG. A única exceção ocorreu na SNR de -15 dB para o algoritmo DE, como



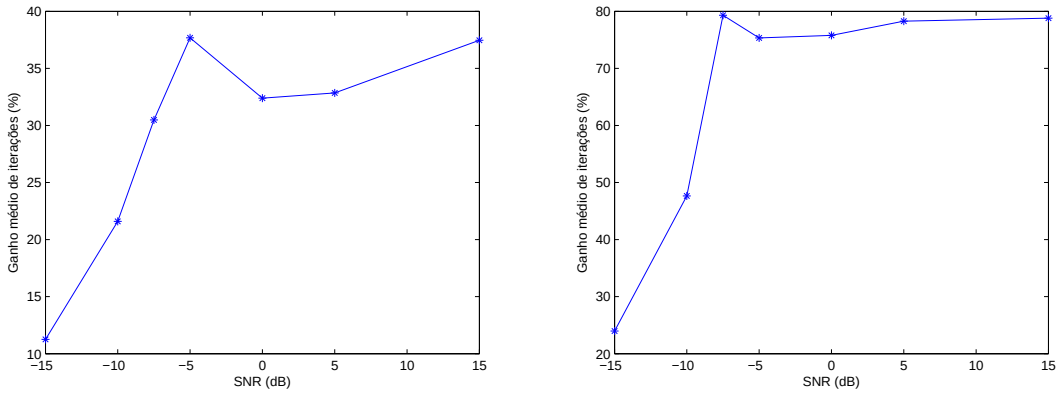
(a) DE (b) CLONALG

Figura 6.4: Percentual de experimentos bem-sucedidos - algoritmos DE e CLONALG.



(a) DE (b) CLONALG

Figura 6.5: Número de avaliações de *fitness* - algoritmos DE e CLONALG.



(a) DE (b) CLONALG

Figura 6.6: Ganho percentual de iterações - algoritmos DE e CLONALG.

mostra a Figura 6.4(a), na qual verificamos uma suave queda no percentual, se comparado com a versão que utiliza uma inicialização uniforme.

Isto, conforme já mencionado na Seção 6.3, pode estar associado ao fato de a resposta em freqüência do filtro privilegiar a região referente aos valores verdadeiros dos ângulos de chegada, em detrimento das restantes. Contudo, especialmente quando a SNR é baixa, o ótimo global pode estar muito distante desta região, de modo que sua posição pode estar associada a uma freqüência na qual o módulo de $H_f(\omega)$ é muito pequeno. Nesta situação, o uso da resposta em freqüência como função densidade de probabilidade na etapa de inicialização pode afastar os indivíduos presentes na população inicial do ótimo global, o que por sua vez, pode ser determinante para que o algoritmo não encontre este ponto.

A Figura 6.5 destaca uma grande vantagem relacionada ao uso da filtragem: nos experimentos em que o ótimo global efetivamente foi localizado, o número de avaliações da função de *fitness* é significativamente reduzido quando a população inicial é amostrada segundo a resposta em freqüência do filtro FIR. Uma vez que tal grandeza nos dá um indicativo do custo computacional do algoritmo, qualquer redução em seu valor, sem que a estimação dos ângulos seja comprometida, é desejável. Neste sentido, o uso do filtro em conjunto com os algoritmos de computação natural se mostra bastante conveniente.

Outro aspecto que podemos observar a partir das curvas mostradas na Figura 6.5 é que, embora tenha ocorrido uma diminuição no número de avaliações da função de *fitness* associado a ambos algoritmos, esta foi mais pronunciada no caso do algoritmo CLONALG. Isto indica que o impacto do uso da filtragem varia de acordo com as características de cada algoritmo.

Por fim, as curvas apresentadas na Figura 6.6 mais uma vez ressaltam os benefícios de se utilizar a filtragem juntamente com os algoritmos de computação natural: é possível observar que os algoritmos, quando utilizam a resposta em freqüência do filtro FIR durante a inicialização da população, encontram o ótimo global em um número de iterações menor, o que, em outras palavras, significa uma convergência mais rápida para este ponto. Além disto, notamos que à medida que o valor da SNR aumenta, este ganho em iterações se torna cada vez mais acentuado.

Isto se deve às variações que a superfície da função ML sofre à medida que a SNR varia. Como ilustrado nas Figuras 2.2 e 2.3, o ótimo global tende a permanecer cada vez mais próximo do ponto associado aos valores verdadeiros dos ângulos de chegada à medida que a SNR aumenta. É evidente, portanto, que nestas situações, os algoritmos DE e CLONALG tenham localizado o ótimo global em um número bem inferior de iterações, pois a população inicial já aloca diversos indivíduos na região próxima ao ponto ótimo, como exemplificado na Figura 6.3.

6.5 Conclusão

Neste capítulo, propusemos uma forma inovadora de gerar a população inicial para os algoritmos de computação natural aplicados ao problema de estimação DOA com o auxílio de um processo de filtragem que visa reduzir a componente de ruído presente na matriz de covariância dos *snapshots*.

Inicialmente, os princípios referentes à filtragem em estimação DOA foram apresentados. Em particular, destacamos que, por intermédio de um filtro FIR de ordem $N-1$, onde N é o número de sensores, e com coeficientes dados pelo autovetor de $\hat{\mathbf{R}}_{\mathbf{y}}$ associado ao seu maior autovalor, a relação sinal-ruído média na saída do filtro é maximizada. Nestas condições, o filtro possui uma resposta em freqüência que, mesmo em condições críticas de relação sinal-ruído, tende a preservar a faixa de freqüências associada aos valores verdadeiros dos ângulos de chegada.

Em seguida, descrevemos a proposta para o uso deste processo de filtragem em conjunto com os algoritmos de computação natural. Em síntese, a idéia é empregar a resposta em freqüência do filtro FIR otimizado como função densidade de probabilidade no processo de amostragem do espaço de busca durante a composição da população inicial. As Figuras 6.2 e 6.3 ressaltam o efeito desta abordagem sobre a distribuição dos indivíduos da população inicial, os quais tendem a se concentrar na região de maior ganho de $H_f(\omega)$, a qual está relacionada aos valores verdadeiros dos ângulos de chegada.

Os resultados obtidos nos experimentos realizados durante a Seção 6.4 enfatizam os benefícios conquistados com o uso desta proposta: em relação ao procedimento padrão de construção da população inicial, o qual baseia-se em uma distribuição uniforme, os algoritmos de computação natural tendem a localizar o ótimo global da função custo J_{ML} com um custo reduzido, expresso tanto em número de avaliações quanto em iterações, quando a resposta em freqüência do filtro FIR é empregada. Além disto, mesmo para valores pequenos de relação sinal-ruído, nos quais a resposta em freqüência do filtro pode desfavorecer a região associada ao ótimo global, observamos que não houve prejuízos significativos no desempenho dos algoritmos.

Diante destas evidências, atestamos que a proposta de uso de filtragem torna os algoritmos de computação natural melhor habilitados para realizar a estimação dos ângulos de chegada.

Capítulo 7

Conclusões e Perspectivas

Neste trabalho, abordamos o problema de estimação de direção de chegada de sinais incidentes em um arranjo de sensores com o auxílio de algoritmos de otimização pertencentes ao ramo de pesquisa denominado Computação Natural.

No Capítulo 2, foram apresentados os princípios referentes ao problema de estimação DOA. Particularmente, destacamos que, empregando o método da máxima verossimilhança, este problema requer a minimização de uma função custo J_{ML} cuja superfície apresenta características desafiadoras, como, por exemplo, a presença de diversos mínimos locais, além do fato de variar à medida que a relação sinal-ruído é alterada.

Devido a estas peculiaridades, a única abordagem capaz de garantidamente encontrar a estimativa ML corresponde a uma busca exaustiva. Porém, esta não pode ser utilizada em virtude de seu elevado custo computacional. Na Seção 2.4, dois métodos clássicos de estimação DOA, a saber, o MODE e o MODEX, foram descritos e analisados. Conforme já apontado na literatura, mostramos que estes métodos perdem qualidade na estimação conforme a SNR diminui, afastando-se do desempenho desejado.

O Capítulo 3 introduziu os fundamentos dos algoritmos de computação natural utilizados neste trabalho. Foram discutidas também as principais razões que sustentam e motivam o uso de meta-heurísticas populacionais para a estimação DOA, à luz das próprias características do

problema e também do teorema *No-free-lunch*.

O passo seguinte consistiu na aplicação dos algoritmos estudados ao problema de estimação DOA. A primeira parte deste estudo foi apresentada no Capítulo 4 e abordou os seguintes aspectos:

1. Ajuste de parâmetros: foram realizados testes de sensibilidade para os principais parâmetros de cada algoritmo. Os resultados obtidos permitiram uma melhor compreensão da influência que eles exercem sobre o comportamento e sobre o desempenho dos algoritmos.

2. Comparação de desempenho no cenário clássico: verificamos que vários dos algoritmos estudados foram capazes de alcançar o desempenho do estimador ML, superando assim os métodos MODE e MODEX, como mostrado na Seção 4.3.

Motivados por estes resultados promissores, passamos então para a segunda parte deste estudo, descrita no Capítulo 5, a qual teve por objetivo investigar o desempenho dos algoritmos de computação natural em outros cenários DOA. Os experimentos realizados forneceram indicativos de que, mesmo nestas novas condições, os algoritmos de computação natural continuam a estimar os ângulos de chegada de maneira adequada.

Por fim, o Capítulo 6 apresentou os fundamentos de um processo de filtragem que visa atenuar o ruído presente nos dados coletados (*snapshots*). Em seguida, introduzimos uma proposta inovadora, baseada nestes conceitos, para a amostragem do espaço de soluções candidatas. Em poucas palavras, o método proposto utiliza a resposta em freqüência do filtro FIR otimizado, isto é, que produz a maior atenuação de ruído em sua saída, como função densidade de probabilidade na geração da população inicial dos algoritmos de computação natural.

Como discutido nas Seções 6.2 e 6.3, ao utilizarmos esta nova abordagem, a expectativa é que a região do espaço de busca situada em torno dos valores verdadeiros dos ângulos de chegada seja privilegiada durante o processo de amostragem, o que por sua vez deve contribuir para um melhor desempenho dos algoritmos de computação natural, podendo, inclusive, torná-los mais

eficientes.

Os resultados obtidos nos experimentos realizados na Seção 6.4 apontam as vantagens desta proposta: quando a resposta em freqüência do filtro FIR é empregada na inicialização da população, os algoritmos de computação natural tendem a localizar o ótimo global da função custo J_{ML} em um número menor de iterações, tendo amostrado e avaliado um número inferior de pontos do espaço de busca, se comparado à situação em que a população inicial é gerada segundo uma distribuição uniforme.

Podemos concluir, portanto, que este trabalho apresentou várias evidências de que o uso de algoritmos de computação natural como alternativa aos métodos clássicos para realizar a estimação dos ângulos de chegada é bastante promissor.

7.1 Perspectivas Futuras

No entanto, é evidente que diversos outros aspectos poderiam ser explorados. Dado que cada vez mais têm surgido novas meta-heurísticas voltadas à otimização de funções, seria pertinente considerar outras ferramentas além das escolhidas para este trabalho. Além disto, diferentes formas de efetuar a análise de desempenho dos algoritmos também poderiam ser adotadas.

Uma perspectiva futura que vislumbramos abordar consiste em empregar o processo de filtragem não apenas na etapa de inicialização da população, quando a resposta em freqüência do filtro FIR otimizado é utilizada como função densidade de probabilidade, mas também para efetivamente filtrar o ruído dos *snapshots*. Neste caso, uma nova versão da função custo teria de ser utilizada.

Por fim, parece-nos interessante avaliar a possibilidade de utilizar este tipo de algoritmo em cenários nos quais os parâmetros DOA variam dinamicamente. Ressaltamos que um passo inicial nesta direção foi dado em (Boccato et al., 2009) e os resultados obtidos naquele trabalho servem como motivação para um estudo mais aprofundado.

Referências Bibliográficas

Ada, G. L., e Nossal, G. J. V. (1987). The clonal selection theory. *Scientific American*, 50–57.

Attux, R. R. F., Loiola, M. B., Suyama, R., de Castro, L. N., Von Zuben, F. J., e Romano, J. M. T. (2003). Blind search for optimal wiener equalizers using an artificial immune network model. *EURASIP Journal of Applied Signal Processing*, *2003*(6), 740–747.

Baker, J. E. (1987). Reducing bias and inefficieny in the selection algorithm. Em J. J. Grefenstette (Ed.), *Int. Conf. on Genetic Algorithms* (págs. 14-21).

Bäck, T., Fogel, D., e Michalewicz, Z. (2000a). *Evolutionary computation 1: Basic algorithms and operators*. Bristol, UK: Institute of Physics Publishing.

Bäck, T., Fogel, D., e Michalewicz, Z. (2000b). *Evolutionary computation 2: Advanced algorithms and operators*. Bristol, UK: Institute of Physics Publishing.

Beyer, H.-G., e Schwefel, H.-P. (2002). Evolution strategies: A comprehensive introduction. *Natural Computing*(1), 3–52.

Boccato, L., de França, F. O., Krummenauer, R., Attux, R., Von Zuben, F. J., e Lopes, A. (2009). Otimização dinâmica imuno-inspirada para o problema de estimação de direção de chegada. Em *Anais do XXVII Simpósio Brasileiro de Telecomunicações.*

Bremermann, H. J. (1962). Optimization through evolution and recombination. Em G. T. J. M. C. Yovits e D. G. Goldstein (Eds.), *Self-organizing systems* (págs. 93–106). Spartan Books.

Brookes, M. (2005). The matrix reference manual. Disponível em `http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/intro.html` (Acessado em 04/08/2010.)

Burnet, F. M. (1959). *Clonal selection theory of acquired immunity*. Cambridge Univ. Press.

Castro, L. N. de, e Timmis, J. (2002). *Artificial immune systems: A new computational intelligence approach*. London, Great Britain: Springer.

Clerc, M., e Kennedy, J. (2002). The particle swarm explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, *6*(1), 58-73.

Cziko, G. (1995). The immune system: Selection by the enemy. Em *Without miracles* (págs. 39-48). MIT Press.

Darwen, P., e Yao, X. (1995). A dilemma for fitness sharing with a scaling function. Em *IEEE International Conference on Evolutionary Computation* (págs. 166–171). Piscataway, NJ.

Darwin, C. R. (1859). *The origin of species*. Wordsworth Editions Limited (1998).

de Castro, L. N. (2006). *Fundamentals of natural computing: Basic concepts, algorithms and applications*. Chapman & Hall/CRC.

de Castro, L. N., e Timmis, J. (2002). An artificial immune network for multimodal function optimization. Em *Proceedings of IEEE Conference on Evolutionary Computation* (Vol. 1, págs. 699-704).

de Castro, L. N., e Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, *6*(3), 239–251.

Deb, K. (1989). *Genetic algorithms in multimodal function optimization*. Dissertação de mestrado, University of Alabama.

Deb, K., e Goldberg, D. E. (1989). An investigation of niche and species formation in genetic function optimization. Em J. D. Schaffer (Ed.), *3rd Int. Conf. Genetic Algorithms* (págs. 42-50).

Forster, P., Larzabal, P., e Boyer, E. (2004). Threshold performance analysis of maximum likelihood DOA estimation. *IEEE Transactions on Signal Processing*, *52*(11), 3183–3191.

Fraser, A. S. (1959). Simulation of genetic systems by automatic digital computers. *Aust. Jour. of Biol. Sci.*, *10*, 489–499.

Friedberg, R. M. (1958). A learning machine: Part I. *IBM Jour. of Research and Development*, *2*, 2–13.

Gehlhaar, D. K., e Fogel, D. B. (1996). Tuning evolutionary programming for conformationally flexible molecular docking. Em L. J. Fogel, P. J. Angeline, e T. Bäck (Eds.), *5th Ann. Conf. on Evolutionary Programming.*

Gershman, A., e Stoica, P. (1999). MODE with extra-roots (MODEX): a new DOA estimation algorithm with an improved threshold performance. Em *IEEE International Conference on Acoustics, Speech, and Signal Processing* (Vol. 5, págs. 2833–2836). Phoenix, AZ.

Gershman, A. B., e Stoica, P. (2000). Data-supported optimization for maximum likelihood DOA estimation. Em *Proc. of the 2000 IEEE Sensor Array and Multichannel Signal Processing Workshop* (págs. 337-341).

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning.* Addison-Wesley.

Goldberg, D. E., e Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. Em *2nd Int. Conf. on Genetic Algorithms* (págs. 41–49).

Haykin, S. (1985). *Array signal processing.* Englewood Cliffs, NJ: Prentice Hall.

Holland, J. (1975). *Adaptation in natural and artificial systems.* University of Michigan Press.

Jerne, N. K. (1974). Towards a network theory of the immune system. *Ann. Immunol. (Inst. Pasteur)*, 373–389.

Kennedy, J. (1997). The particle swarm: social adaptation of knowledge. *IEEE International Conference on Evolutionary Computation*, 303–308.

Kennedy, J. (2004). Particle swarms: Optimization based on sociocognition. Em L. N. de Castro e F. J. von Zuben (Eds.), *Recent Developments in Biologically Inspired Computing* (págs. 235–269). Idea Group Publishing.

Kennedy, J., e Eberhart, R. (1995). Particle swarm optimization. *IEEE International Conference on Neural Networks*, *4*, 1942–1948.

Kennedy, J., Eberhart, R., e Shi, Y. (2001). *Swarm intelligence.* Morgan Kaufmann Publishers.

Krim, H., e Viberg, M. (1996). Two decades of array signal processing research: the parametric approach. *IEEE Signal Processing Magazine*, *13*(4).

Krummenauer, R. (2007). *Filtragem ótima na estimação de direção de chegada de ondas planas usando arranjo de sensores*. Dissertação de mestrado, Faculdade de Engenharia Elétrica e Computação - UNICAMP, Campinas, SP.

Krummenauer, R., Cazarotto, M., Lopes, A., Larzabal, P., e Forster, P. (2010). Improving the threshold performance of maximum likelihood estimation of direction of arrival. *Signal Processing*, *90*(5), 1582-1590.

Kumaresan, R., Scharf, L. L., e Shaw, A. K. (1986). An algorithm for pole-zero modeling and spectral analysis. *IEEE Trans. on Acoustics, Speech and Signal Processing*, *ASSP-34*(3).

Mahfoud, S. W. (1995a). *Niching methods for genetic algorithms*. Tese de doutorado, University of Illinois at Urbana-Champaign.

Mahfoud, S. W. (1995b). Population size and genetic drift in fitness sharing. Em L. D. Whitley e M. D. Vose (Eds.), *Proc. Foundations Genetic Algorithms* (págs. 185-223).

Manikas, A. (2004). *Differential geometry in array processing*. Imperial College Press.

Michalewicz, Z. (1996). *Genetic algorithms + data structures = evolution programs* (3ª ed.). Springer-Verlag.

Michalewicz, Z., e Fogel, D. B. (2004). *How to solve it: Modern heuristics* (2ª ed.). Springer.

Moscato, P. (1989). *On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms* (Relat. Técnico). Caltech Concurrent Computation Program.

Papoulis, A., e Pillai, S. U. (2002). *Probability, random variables and stochastic processes* (4ª ed.). McGraw-Hill Companies.

Price, K., Storn, R., e Lampinen, J. A. (2005). *Differential evolution: A practical approach to global optimization*. Springer.

Pétrowski, A. (1996). A clearing procedure as a niching method for genetic algorithms. Em *IEEE International Conference on Evolutionary Computation* (págs. 798–803).

Rechenberg, I. (1965). Cybernetic solution path of an experimental problem. *Royal Aircraft Establishement Library Translation 1122*.

Rechenberg, I. (1971). *Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biologischen evolution*. Tese de doutorado, Technical University of Berlin.

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biologischen evolution*. Frommann-Holzboog.

Rife, D., e Boorstyn, R. (1974). Single tone parameter estimation from discrete-time observations. *IEEE Transactions on Information Theory*, *20*(5), 591–598.

Rudolph, G. (1992). On correlated mutations in evolution strategies. Em R. Männer e B. Manderick (Eds.), *Parallel problem solving from nature - seconf. conf. PPSN* (págs. 105–114).

Sareni, B., e Krahenbuhl, L. (1998). Fitness sharing and niching methods revisited. *IEEE Transactions on Evolutionary Computation*, *2*, 97–106.

Schmidt, R. O. (1981). *A signal subspace approach to multiple emitter location and spectral estimation*. Tese de doutorado, Stanford University.

Schmidt, R. O. (1986). Multiple emitter location and signal parameter estimation. *IEEE Trans. Antennas Propag.*, *AP-34*(3).

Schwefel, H.-P. (1965). *Kybernetische evolution als strategie der exprimentellen forschung in der strömungstechnik*. Dissertação de mestrado, Technical University of Berlin.

Schwefel, H.-P. (1977). *Numerische optimierung von computer-modellen mittels der evolutionsstrategie*. Birkhäuser.

Schwefel, H.-P. (1981). *Numerical optimization of computer models*. New York, NY, USA: John Wiley & Sons, Inc.

Schwefel, H.-P. (1995). *Evolution and optimum seeking*. Wiley.

Sharman, K. C. (1988). Maximum likelihood parameter estimation by simulated annealing. Em *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (págs. 2741-2744).

Sharman, K. C., e McGlurkin, G. D. (1989). Genetic algorithms for maximum likelihood parameter estimation. Em *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*

(págs. 2716-2719).

Shi, Y., e Eberhart, R. C. (1999). Empirical study of particle swarm optimization. Em *IEEE Congress on Evolutionary Computation* (págs. 1945–1950).

Stoica, P., e Gershman, A. B. (1999). Maximum-likelihood DOA estimation by data-supported grid search. *IEEE Signal Processing Letters*, 273-275.

Stoica, P., e Nehorai, A. (1990). Performance study of conditional and unconditional direction-of-arrival estimation. *IEEE Trans. on Acoustics, Speech and Signal Processing*, *ASSP-38*(10).

Stoica, P., e Sharman, K. C. (1990). Novel eigenanalysis method for direction estimation. *IEE Proceedings part F (Radar and Signal Processing)*, *137*(1).

Storn, R., e Price, K. (1995). *Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces* (Relat. Técnico Nº TR-95-012). ICSI.

Storn, R., e Price, K. (1997). Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*, 341-359.

Van Trees, H. L. (2001a). *Detection, estimation, and modulation theory - part I.* John Wiley & Sons.

Van Trees, H. L. (2001b). *Optimum array processing. part IV of detection, estimation and modulation theory.* New York, USA: John Wiley & Sons.

White, T., e Pagurek, B. (1998). Towards multi-swarm problem solving in networks. Em *3rd int. conf. on multi-agents systems (ICMAS)* (págs. 333–340).

Wolpert, D. H., e Macready, W. G. (1995). *No free lunch theorems for search* (Relat. Técnico Nº SFI-TR-95-02-010). Santa Fe Institute.

Wolpert, D. H., e Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(2), 67-82.